Secure Signal Processing:

Privacy Preserving Cryptographic Protocols for Multimedia

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. ir. K. C. A. M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op donderdag 24 juni om 15:00 uur
door Zekeriya ERKIN
Master of Science in Computer Engineering, Istanbul University of Technology,
geboren te Nizip, Turkije.

Published by TU Delft Mediamatica
Printed by Wöhrmann Print Service
ISBN: 978-90-813811-9-2
Cover design by Beste Ertürk

Secure Signal Processing:

Privacy Preserving Cryptographic Protocols for Multimedia

Preface

The research for this thesis was conducted within the Signal Processing in the Encrypted Domain (SPEED) project which was funded by the European Union within the Sixth Programme Framework. It started in December 2006 and finished in December 2009. The goal of the project was to foster the advancement of the marriage between Signal Processing and Cryptographic techniques, both at theoretical and practical level. The objective was the initiation and development of a totally new and unexplored interdisciplinary framework and technologies for signal processing in the encrypted domain that address the problem of security in multimedia communication/consumption, and digital signal manipulation.

In this European project, the following parties were involved: Università degli Studi di Siena, Università degli Studi di Firenze, Katholieke Universiteit Leuven, Ruhr-Universitaet Bochum, Philips Electronics Nederland B.V. and Delft University of Technology.

During the SPEED project, security and privacy problems in several multimedia applications were addressed and a set of solutions that merges cryptography and signal processing was proposed. In this thesis, we present the results of the research conducted in Delft University of Technology.

Z. Erkin, Delft, December 2009.

Table of Contents

List of Figures

List of Tables

One

Introduction

“. . . the right to be let alone.”
Warren and Brandeis

Today, we are witnessing one of the most important breakthroughs in history. Started in the form of electronic mails, text messaging and World Wide Web, we have created a virtual world that has the advantage of accessibility from any place at any time and offers almost unlimited variety of services unlike any of its physical counterparts. In this virtual world, people can access information and knowledge instantly, create groups to share and discuss ideas, do shopping, entertain themselves and much more. As the advantages of the virtual world are undisputed, more functions from the real world are brought to the virtual one, resulting in an increasingly connected world.

A close look at the services today shows that most of the services rely on data processing. Typical data for an online shopping site would be the identifiers, properties and the quantities of the products on sale. For a social network site personal data such as likes and dislikes would be considered. Regardless of the application type, most of the services rely highly on data collected from the users to design better applications in terms of service experience. As an example, for a shopping site, it can be very helpful to show the most popular products on the first page of the web site. Of course, to add such a functionality, the service provider needs to record and process the shopping patterns of the users. To make the system even more attractive, these sites can offer personalization. Depending on demographic properties, preferences and past actions, the service provider can generate specific recommendations that specific users may like.

Despite the fact that a more connected world simplifies people’s lives by providing several services, the available information on the users of online applications creates a serious privacy risk for the users. Every piece of data collected contains sensitive information about the users that can be abused by other parties including the service provider [14]. In addition to the privacy consideration of the users, the service provider may have his own concerns for securing his service against malicious users who may try to abuse the service for their own benefit. In either case, we face a challenging problem in the virtual world in which involved parties do not fully trust each other with their sensitive data.

1.1 Case Scenario: Automated Medical System

To illustrate the problem of lack of trust, consider the example of patient-doctor relation. During a medical examination, the patient *trusts* his doctor on the confidentiality of the examination result. The patient-doctor relation is built on the strong assumption that the doctor will keep his Hippocratic Oath. Now, imagine that due to the lack of doctors and increasing number of patients in society, an automated system for medical diagnosis is to be deployed. This system consists of an expert system with a large database of recordings on diseases and their symptoms. Regularly or on demand, a device given to the patient makes some measurements on the patient and sends its data to the central system where the expert system tries to make a diagnosis. Depending on the analysis, the expert system may suggest different things such as conducting another set of analysis, making an appointment at a hospital or even prescribing medicine.

In this scenario, both sides, the patient and the service provider, have several advantages. The patient can have medical check-ups at any time and at any place, eliminating a tedious procedure of making appointments with the doctor. At the same time, the service provider can keep the expert system online without difficulty and serve a lot more people concurrently. In general, the whole medical system can benefit from reducing the expenses, saving time and valuable resources. The question fundamental to this thesis is whether we can move the trust model between the doctor and the patient to the virtual world.

A straightforward approach to secure this medical system usually considers the confidentiality of the communication channel and the stored data. These precautions may prevent attackers from obtaining highly privacy-sensitive data. However, the real privacy threat in this scenario arises from the fact that it is not a valid assumption for the patient to fully trust the service provider with his medical records. The service provider may have an interest in collecting information on the patients since this type of data can be particularly interesting for insurance companies or employers. In the case of misuse, the consequences will be severe for the privacy of the patient.

In summary, in online applications where the service provider and the user interact virtually, the involved parties may have sensitive data that they would like to keep secret from the other parties. For instance in dating sites as a social network, the service provider finds similar other users based on user's preferences. In the case of online shopping sites, the service provider tries to increase his profit by providing targeted advertisements by observing user's shopping behavior and/or profile. In either case, the service provider needs to access the privacy-sensitive data of the users. As this constitutes a serious privacy risk for the users, some users may not prefer to use the service at all [15, 17]. And for those who choose to get the service, which has no proper privacy protection, are open to privacy breaches. The situation can get worse as in the case of surveillance systems in which the users are being monitored without their consent [24].

1.2 Preserving Privacy

In order to protect the privacy of the patient, we consider two different solutions: using a trusted third party (TTP) and secure signal processing. Using the medical system scenario as an example, we illustrate how a TTP can be coupled to a privacy-preserving solution. In such a setting, the TTP, who is trusted by all parties, receives the privacy-sensitive data from the patient and the algorithm from the service provider. The TTP can either run the algorithm by using the private data of the patient and report the outcome himself or he can anonymize the patient's data and give them to the server to be processed. In either case, the security and the privacy concerns are eliminated as the patient's privacy sensitive data and the algorithm are safe in the hands of the TTP. The problem with this approach is that in real life it is not easy, if not impossible, to find TTPs that do not have motives of business, politics, etc. In business, it is strongly believed that TTPs are vulnerable, costly and risky [22]. Thus, instead of giving away the privacy-sensitive data and the algorithm to a TTP, we can explore cryptographic techniques.

A solution based on cryptographic techniques would be as follows. The device provided to the patient makes the measurements, encrypts the data and sends it to the automated system. Upon receiving the data, the automated medical system runs its algorithm on the encrypted data and obtains the diagnosis result, again in encrypted form. The encrypted diagnosis is then sent to the patient; he decrypts the encrypted message and obtains the diagnosis (Fig. 1.1). As a consequence, the patient does not reveal his medical data to the automated system but obtains the diagnosis which is in turn unknown to the automated system.

In the medical scenario we assume that each party plays his own role properly. That is, the steps defined by the protocol are followed and no manipulation either on the data or in the algorithm is made. This type of model, known as *semi-honest* model, also expects the parties to record the previous messages in order to deduce more information than they are supposed to have. In the case of manipulating the data or the steps of the protocol, extra precautions should be taken to ensure the correctness of the protocol. These precautions usually consist of cryptographic protocols such as zero-knowledge proofs in which one party tries to prove to another that a statement is true without revealing the statement itself [12]. This security model is often referred to as *malicious case* or *active adversary* model [12]. Throughout this thesis, we assume that all parties act according to the semi-honest model.

1.3 Signal Processing and Cryptographic Tools

The proposed system based on cryptographic techniques in automated medical system scenario provides the necessary privacy protection for the patient. However, realizing the system described in Fig. 1.1 presents a number of challenges. The goal of encryption is to make the original message unreadable in such a way that only the recipient of the message with the right key can read it. After the encryption, the structure of the original message is destroyed and the resulting cipher text looks totally random. As a result, once the message is encrypted, operations on them such as sorting and

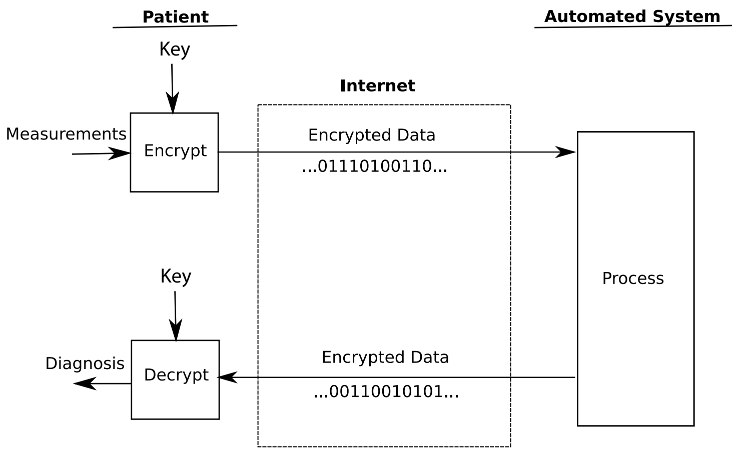


Figure 1.1: Privacy-preserving medical diagnosis system.

averaging a set of encrypted values become non-trivial. Therefore, we need to deploy cryptographic protocols to process data in the encrypted domain.

Before describing existing cryptographic tools for processing encrypted data, we need to identify what kind of processing is required in online multimedia applications in general. A wide variety of services available on the Internet today possess similar features. The data in question is usually a set, or more precisely a vector, of values that might be preferences of users (social network sites), likes or dislikes (recommender systems) and media files (audio, image and video). The service provider processes the data depending on the service demanded. As an example, in the case of social networks, the focus is on finding the most similar users based on their preferences. In the case of recommender systems, the service provider first needs to find the most similar users and then generate recommendations by applying some statistical methods like averaging similar users' ratings. Many other examples can be given here such as finding other copies of a picture or matching the face picture of a user to a celebrity.

In all of the applications mentioned above and considered in this thesis, we see that the data possess the structure of signals, that is they are correlated values from a small range, and the applications consist of common operations from the field of signal processing such as averaging and quantization. Even though the classification of signal processing operations is out of the scope of this thesis [3], the operations we observe in multimedia applications can be grouped in two primary categories:

- **Linear operations:** This group consists of operations such as linear transforms, correlation, linear filtering, computation of difference and error signals.
- **Non-linear operations:** Distance computation, comparison, thresholding and quantization can be named here as examples.

Considering that the data in multimedia applications are privacy-sensitive and we propose to ensure the confidentiality of the data by means of encryption, we need to realize the linear and non-linear parts of a signal processing application under encryption. In order to process encrypted data, we can exploit *homomorphic encryption schemes* and *secure multi-party computation* (MPC) techniques.

1.3.1 Linear Operations and Homomorphism

In cryptography, a number of public key cryptosystems possess a property called *homomorphism* such that after encrypting a message, there is some structure preserved that can be exploited to process it in the encrypted domain [1]. In particular, this means that an operation on the encrypted data corresponds to another operation on the plain text. For instance, the multiplication of two encryptions with a *multiplicatively* homomorphic cryptosystem like RSA [21] gives us the encrypted product of these messages:

$$D_{sk}(E_{pk}(m_1) \times E_{pk}(m_2)) = m_1 \times m_2, \tag{1.1}$$

where m_1, m_2 are messages and, E_{pk} and D_{sk} correspond to encryption and decryption functions with the public and the secret key, respectively. A second type of homomorphism allows us to have the encrypted sum of messages when multiplied in the encrypted domain. This property is called *additive* homomorphism:

$$D_{sk}(E_{pk}(m_1) \times E_{pk}(m_2)) = m_1 + m_2, \tag{1.2}$$

where E_{pk} and D_{sk} are defined as before but for an additively homomorphic cryptosystem like Paillier [18]. As a consequence of additive homomorphism, a message can be multiplied with a public constant c by raising the encryption of the message to the power of that constant:

$$D_{sk}(E_{pk}(m)^c) = m \cdot c. \tag{1.3}$$

Depending on the particular cryptosystem used, addition or multiplication[1] can be carried out on encrypted values. This allows us to realize linear operations in the encrypted domain. As an example, consider that the similarity of two users, A and B, is to be calculated in a recommender system. Assume that each user is represented by his preference vector V_A and V_B, respectively. In order to obtain the similarity value, the inner product of V_A and V_B is needed. This inner product computation can be realized in a secure way as follows: user A encrypts his vector V_A and sends it to user B. Upon receiving V_A, user B computes the inner product by using the additive homomorphism property of the cryptosystem as shown below:

[1]Recently, fully algebraic cryptosystems were proposed in [2] and [11] based on polynomials and lattices, respectively. However they are highly inefficient to be used in practice but very important to prove the existence of such cryptosystems.

$$\begin{aligned} E_{pk_A}(< V_A, V_B >) &= E_{pk_A}(\sum_{i=1}^{N} V_{A,i} \cdot V_{B,i}) \\ &= E_{pk_A}(V_{A,1} \cdot V_{B,1} + \ldots + V_{A,N} \cdot V_{B,N}) \\ &= E_{pk_A}(V_{A,1})^{V_{B,1}} \cdot \ldots \cdot E_{pk_A}(V_{A,N})^{V_{B,N}} \\ &= \prod_{i=1}^{N} E_{pk_A}(V_{A,i})^{V_{B,i}}, \end{aligned} \tag{1.4}$$

where $< V_A, V_B >$ represents the inner product of user vectors V_A and V_B. In other words, the inner product of one encrypted and one plain vector can be calculated with multiplications and exponentiations in the encrypted domain.

1.3.2 Non-linear Operations and MPC

In the case of non-linear operations, homomorphic property is not sufficient. In such cases, *secure multiparty computation* (MPC) techniques known from cryptography must be used [25]. These techniques allows to evaluate a function with secret inputs from a number of parties such that each party will only know its own contribution and the intended result of the function.

The field of MPC and its sibling secure function evaluation is old and many positive results have been published [4, 13, 26]. In literature, we see that MPC can be based on different techniques ranging from circuit scrambling to secret sharing and public-key cryptosystems. In all of these techniques, the idea is the evaluation of a circuit either Boolean or arithmetic over some field or ring. In the case of circuit approach with two players A and B, a function f with secret inputs from both parties is constructed as a Boolean circuit by user A. Each wire of every gate is associated with two keys, one key for bit value 1 and another key for bit value 0. The keys are used to construct the truth tables. Then, the shuffled truth tables are sent to user B. In order to evaluate the function f, user B also needs to know the input of user A. To obtain his inputs, user B initiated an Oblivious Transfer (OT) protocol [12]. OT protocols allows B to acquire the correct input for each wire without revealing his input to user A. Together with his input bits and the oblivious transfer of user A's bits, user B can evaluate the Boolean circuit and obtain the result.

While Boolean circuits for any function f can be constructed easily, the size of the circuit plays an important role for the efficiency. In the case of complex functions and operations like multiplication, the size of the circuit grows dramatically. As size grows, the construction and the evaluation of the function become more cumbersome. As most real-world applications are infeasible to rephrase with a Boolean circuit due to the required size of the circuit, we do not consider this approach in this thesis. Instead, we focus on evaluation of circuits over integers and use the term MPC in that context.

To illustrate the role of MPC in realizing non-linear operations in the encrypted domain, assume that party A would like to compute the minimum squared Euclidean distance of his his vector V_A to one of the K vectors in an R dimensional space. User

A and B want to keep their vectors secret. The squared Euclidean distance for the two vectors V_A and V_B^j, for $j = 1$ to K is:

$$\begin{aligned} D^2(V_A, V_B^j) &= \sum_{i=0}^{R-1} (v_{A,i} - v_{B,i}^j)^2 \\ &= \sum_{i=0}^{R-1} v_{A,i}^2 - 2 \cdot v_{A,i} \cdot v_{B,i}^j + (v_{B,i}^j)^2. \end{aligned} \tag{1.5}$$

Imagine that user B provides the encrypted inputs $E_{pk_B}(v_{B,i}^j)$ and $E_{pk_B}((v_{B,i}^j)^2)$. Then, the squared Euclidean distances can be computed by user A as follows:

$$\begin{aligned} E_{pk_B}(D^2(V_A, V_B^j)) &= E_{pk_B}(\sum_{i=0}^{R-1} (v_{A,i} - v_{B,i}^j)^2) \\ &= \prod_{i=0}^{R-1} E_{pk_B}(v_{A,i}^2) \cdot E_{pk_B}(v_{B,i}^j)^{-2 \cdot v_{A,i}} \cdot E_{pk_B}((v_{B,i}^j)^2). \end{aligned} \tag{1.6}$$

Notice that the first and the second terms can be computed by user A, while the third term is provided directly by user B. After having computed K squared distances, user A has to find out the minimum of these values. As he does not possess the decryption key, he cannot observe the contents of the encryptions and decide himself. Since finding the minimum requires comparison and it is not a linear operation, user A and B need to run a cryptographic protocol based on MPC techniques [26] to compare the encrypted squared distances. Unlike the straightforward application of homomorphism property, MPC techniques are interactive and involve usually complicated protocols. Depending on the function to be implemented, time, computation power, bandwidth and storage space requirements can be demanding.

1.4 Problem Statement

As illustrated by the medical system scenario, severe privacy threats in online multimedia applications exist. This problem cannot be solved by deploying secure channels or keeping privacy-sensitive data of the users encrypted on the server side. While these security measures eliminate a number of security threats from outside attackers, they are not sufficient to protect the sensitive data against misuse by the service provider which creates the biggest potential risk.

In this thesis, we focus on principled solutions to protect the privacy of users in multimedia applications. For this purpose we propose to keep the privacy-sensitive data safe by means of encryption during processing. This approach eliminates the risk of possible privacy abuses as the sensitive data is only available to the owner but not to the other parties. However, once encrypted, the structure in data is destroyed as a consequence of the encryption procedure. In order to process encrypted data, we investigate cryptographic tools such as homomorphism and MPC techniques.

The homomorphism is typically used for implementing linear operations but it is not sufficient for developing non-linear operations. MPC techniques, being inefficient for realizing linear operations, provide a basis to implement the non-linear parts of an application. However, these techniques are mostly generic and do not consider the signal aspects of the application and thus, if they are applied directly, the result will be costly in terms of time, computation power, bandwidth requirement or storage capacity. Therefore, this thesis focuses on solutions for preserving privacy in multimedia applications by introducing a new idea, to the best of our knowledge, which proposes using cryptographic tools that exploit the signal processing aspects of the application.

In order to illustrate the idea of the integration of signal processing and cryptography, we have selected prototypical applications. In particular, we focus on face detection, clustering, recommender systems and digital content fingerprinting. These applications are selected as they consist of common signal processing operations such as scaling, correlation, distance computation, thresholding and finding minimums which can be seen in other multimedia applications too. In order to realize privacy preserving version of multimedia applications, such operations should be realized in the encrypted domain efficiently. To achieve this goal, we have addressed the following challenges:

- data representation,
- realizing linear and non-linear operations in the encrypted domain,
- data expansion due to encryption,
- communication and computation costs of using cryptographic protocols.

After presenting cryptographic tools that are related to our purpose and signal aspects of multimedia applications in Chapter 2, a more formal problem statement will be given in Section 2.5.

1.5 Thesis Outline

This thesis is organized to cover all aspects of the selected prototypical applications. In order to have a clear view on the available cryptographic tools and existing solutions that address similar problems, we start with an overview chapter. The overview is followed by a number of chapters each of which concentrates on one particular multimedia application and presents a complete solution. We finalize the thesis with a discussion that summarizes what has been achieved and which challenges require further research.

Chapter 2
Protection and retrieval of encrypted multimedia content: When cryptography meets signal processing

A new approach to design privacy preserving multimedia applications that merges cryptography and signal processing requires an understanding of both disciplines. As cryptography is not a familiar subject in signal processing community, we start Chapter 2 with a brief introduction to cryptographic tools that can be used in designing cryptographic protocols and discuss the security requirements in privacy-preserving signal processing applications. In order to illustrate the use of the cryptographic tools, we summarize related work in the field for a number of selected applications. Chapter 2, which has been published as “Protection and retrieval of encrypted multimedia content: When cryptography meets signal processing” by Z. Erkin, A. Piva, S. Katzenbeisser, R. L. Lagendijk, J. Shokrollahi, G. Neven, and M. Barni in *Eurasip Journal on Information Security*, 20 pages, 2007, ends with the formal problem statement of this thesis.

Chapter 3
Privacy-Preserving Face Recognition

Identification systems based on biometric data have become increasingly important for commercial use. In this chapter we consider surveillance systems as an example and investigate its privacy aspects. Such systems play a crucial role in providing security as they enable authorities to monitor physical locations in real time and thus, they are deployed in vast numbers. It is also possible to misuse surveillance systems for tracking and locating purposes as they cover almost every major highway, street and square. Therefore, we propose a solution based on cryptographic techniques that can be used to hide the face image of a person captured by the camera but still permits to check if that person has a record in a remote database. The protocol we propose is based on Eigenface algorithm [23] that finds the most similar person in the database. However, instead of an image in the clear, our protocol accepts an encrypted image. This significant change in the setting introduces challenges in the detection algorithm which requires to realize signal processing operations such as projection, distance computation, minimum distance computation and thresholding in the encrypted domain. The proposed solution for the surveillance system, in particular face detection, can be generalized to many other signal processing applications. This chapter is an integral copy of “Privacy-preserving face recognition” by Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, R. L. Lagendijk, and T. Toft in the *9th Symposium on Privacy Enhanced Technologies (PETs)*, pages 235–253, 2009.

Chapter 4
Privacy-Preserving User Clustering in a Social Network

A very common application on the Internet is finding similar people in social networks. As the purpose of the social networks may change from dating to finding people with the same disease, users of such social networks may not want to reveal their highly privacy-sensitive data to others and to the service provider. In Chapter 4, we address this problem and propose a way to find similar users in a social network without revealing user preferences. The solution is based on widely used K-means clustering algorithm [10] where people are assigned to the most similar group. Here, we propose a method based on secure multiparty computation techniques to realize the steps of K-means algorithm such as computing distances to the existing cluster centroids, finding the closest cluster and updating the centroids when the user's data are encrypted. This chapter is an integral copy of "Privacy-preserving user clustering in a social network" by Z. Erkin, T. Veugen, T. Toft, and R. L. Lagendijk in the *First IEEE Workshop on Information Forensics and Security (WIFS09)*, pages 96–100, 2009.

Chapter 5
Privacy-Preserving Recommender System

Getting recommendations has become very common for online services such as shopping, traveling, dating, etc. Such services generate recommendations based on user information which can be obtained from user's demographic information, preferences and past actions. As the information collected by the system can be abused by the service provider, the protection of the data is necessary. In Chapter 5, we propose a solution for recommender systems that can generate the required recommendation by using encrypted ratings of users. In this system, the service provider does not get information on its users whereas the users can get accurate recommendations. This chapter is an integral copy of "Privacy-preserving centralized recommender system" by Z. Erkin, T. Veugen, T. Toft and R. L. Lagendijk in the *IEEE Transactions on Information Forensics and Security*, (in preparation) 2010.

Chapter 6
Anonymous Fingerprinting

Similar to the trust problem between the service provider and the users in the applications presented in the previous chapters, a digital content buyer may have problems in trusting the seller. In general, the seller of a digital content protects himself by embedding a watermark in the content. In this way, he can prove his ownership of the content during a dispute. In order to identify the source of illegal distribution, he can also embed the identity of the buyer. This approach, also known as fingerprinting, has the disadvantage that the seller possesses the fingerprint of the buyer in clear. Having the fingerprint of the buyer in clear, the seller can embed it into any digital content without the buyer knowing it and accuse him for illegal distribution later on. To eliminate this threat, anonymous fingerprinting protocols were developed based on cryptographic tools such as homomorphic cryptosystems and zero-knowledge proof protocols [16]. However, despite the security and correctness of the proposed protocols, the underlying watermarking system is vulnerable even to the simplest attacks. In Chapter 6, we propose to adapt state-of-the-art watermarking schemes robust against several types of attacks and address the problems of working in the encrypted domain. This chapter is an integral copy of "Anonymous fingerprinting with robust QIM watermarking techniques" by J. P. Prins, Z. Erkin, and R. L. Lagendijk in the *Eurasip Journal on Information Security*, 2007:1–7, 2007.

Chapter 7
Conclusion and Discussions

Considering the multimedia applications and solutions presented in previous chapters, in Chapter 7 we summarize proposed solutions that combine cryptography and signal processing to develop privacy-preserving multimedia applications. Since we are interested in principled solutions for preserving privacy in multimedia applications, this chapter discusses the common approaches in our proposed solutions and connects the pieces from each chapter to form an understanding on the general problem of working in the encrypted domain. We analyze what has been achieved regarding the problems stated in Chapter 2 and we conclude discussing which problems still require further research.

1.6 Contributions

This thesis focuses on principled solutions to protect the privacy in multimedia applications and thus, a number of prototypical applications were selected to identify the challenges for processing encrypted signals. Several contributions have been made:

- For the first time, to the best our knowledge, the idea of processing encrypted data within the context of signal processing is addressed that aims for better efficiency in terms of computational complexity and bandwidth requirements such that the proposed solutions can be considered to be deployed in real life [6, 7]. To achieve this goal, the following major challenges are addressed:
  - **Data representation.** The applications we consider are from the field of signal processing and thus, they operate on signal values. These signal values can be integer values in the beginning like pixel values of an image but they mostly become real values after processing. The bit length of the values can also change depending on the operation. Unfortunately, currently most of the existing homomorphic cryptosystems work on integer values. Thus, we propose a strategy for data representation for working in the encrypted domain that copes with real values and possible expansion in bit length of signals throughout the processing.
  - **Linear Operations and Homomorphism.** The homomorphism property of the public key cryptosystems is exploited for designing the linear parts of privacy-preserving multimedia applications [5, 8, 9, 19, 20]. In particular, scaling, projection and correlation computations are realized by using homomorphism property given that one of the inputs of the computation such as scaling factor is known in plain. In such a case, the required output can be computed by one party by carrying out multiplications and exponentiations on the encrypted data. We address several linear operations for different settings and propose methods to realize the operations with minimum overhead.
  - **Non-linear operations and MPC.** Realizing non-linear operations with encrypted data is a challenging task as it requires to design cryptographic

protocols based on MPC techniques. In [5, 8, 9], we propose a number of cryptographic protocols for several non-linear operations including distance computation, thresholding and comparison. The proposed solutions differ significantly depending on the setting. In distance computation of two user vectors, for instance, the homomorphism property is sufficient for the computations of the linear parts. For the squared term, there is no interaction needed as it can be computed and sent in the beginning of the protocol [8]. However, if the vectors are both encrypted and should be kept secret from the owner of the decryption key, homomorphism property is not sufficient alone and running a cryptographic protocol is necessary [5]. The proposed cryptographic protocols for such cases, which are based on homomorphism and MPC techniques, are particularly developed for the signal processing applications to achieve better efficiency in terms of computational and communication costs compared to existing solutions that use generic cryptographic tools.

- **Data expansion.** Since we use semantically secure cryptosystems, the data expansion after encrypting a signal value, which is much smaller compared to the key size of the encryption scheme, constitutes a major drawback for the storage and transmission of the encrypted data. In addition, we deploy interactive cryptographic protocols to realize non-linear operations which increase the bandwidth requirement further. This problem is addressed in [9] and an effective solution, namely data packing, is proposed to be used. Instead of encrypting individual signal samples, we pack a number of them in one encryption and process the packed data later on. As a consequence, the cryptographic protocols for processing the encrypted data are modified to reflect the change in the construction. Data packing considerably reduces the communication and computational costs since less number of encryptions are transmitted and processed.
- **Computational costs.** The realization of signal processing operation in the encrypted domain introduces overhead in terms of computation power and bandwidth requirements compared to the original systems in plain. For instance finding the minimum of a thousand values can be done in a few microseconds but a similar operation with a thousand encrypted values takes time in the order of minutes. The challenge of minimizing the computation power is addressed in [5, 8, 9] which focus on designing the cryptographic protocols that minimize the number of operations on the encrypted data.

References

[1] N. Ahituv, Y. Lapid, and S. Neumann. Processing encrypted data. *Commun. ACM*, 30(9):777–780, 1987.

[2] F. Armknecht and A.-R. Sadeghi. A new approach for algebraically homomorphic encryption. Cryptology ePrint Archive, Report 2008/422, 2008. `http://eprint.iacr.org/`.

[3] M. Barni. Preliminary report on s.p.e.d. theory. www.speedproject.eu.

[4] D. Chaum, C. Crépeau, and I. Damgard. Multiparty unconditionally secure protocols. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 11–19, New York, NY, USA, 1988. ACM.

[5] Z. Erkin, M. Franz, S. Katzenbeisser, J. Guajardo, R. Lagendijk, and T. Toft. Privacy-preserving face recognition. In *9th Symposium on Privacy Enhanced Technologies (PETs)*, pages 235–253, Seattle, USA, August 2009.

[6] Z. Erkin and R. L. Lagendijk. On processing encrypted data. In *13th annual conference of the Advanced School for Computing and Imaging*, pages 322–329, June 13-15 2007.

[7] Z. Erkin, A. Piva, S. Katzenbeisser, R. L. Lagendijk, J. Shokrollahi, G. Neven, and M. Barni. Protection and retrieval of encrypted multimedia content: When cryptography meets signal processing. *Eurasip Journal on Information Security*, 2007, Article ID 78943, 20 pages, 2007.

[8] Z. Erkin, T. Veugen, T. Toft, and R. Lagendijk. Privacy-preserving user clustering in a social network. In *1st IEEE Workshop on Information Forensics and Security (WIFS09)*, pages 96–100, 2009.

[9] Z. Erkin, T. Veugen, T. Toft, and R. L. Lagendijk. Privacy-preserving centralized recommender system. *IEEE Transactions on Information Forensics and Security*, (in preparation), 2010.

[10] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, 1990.

[11] C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC: Proceedings of the 41st annual ACM Symposium on Theory of Computing*, pages 169–178, 2009.

[12] O. Goldreich. *Foundations of Cryptography I*. Cambridge University Press, 2001.

[13] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC '87: Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 218–229, New York, NY, USA, 1987. ACM.

[14] A. Gregory. Data abuse is a rapidly growing problem, November 2008. http://www.securitypark.co.uk/security_article262328.html.

[15] R. Jennings. European social technographics revealed, February 2008.

[16] M. Kuribayashi and H. Tanaka. Fingerprinting protocol for images based on additive homomorphic property. *IEEE Transactions on Image Processing*, 14(12):2129–2139, December 2005.

[17] Ofcom. Social networking: A quantitative and qualitative research report into attitudes, behaviours and use, April 2008.

[18] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology—EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.

[19] J. P. Prins, Z. Erkin, and R. L. Lagendijk. Anonymous fingerprinting with robust QIM watermarking techniques. *Eurasip Journal on Information Security*, 2007:1–7, 2007.

[20] J. P. Prins, Z. Erkin, and R. L. Lagendijk. Robust anonymous fingerprinting. In *28th Symposium on Information Theory in the Benelux*, pages 59–66, May 24-25 2007.

[21] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[22] N. Szabo. Trusted third parties are security holes, 2005. http://szabo.best.vwh.net/ttps.html.

[23] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, pages 586–591, 1991.

[24] R. Walden. Surveillance and super databases: New privacy threats in the information and technology age, September 2007. http://humanrights.suite101.com/article.cfm/under_the_eye_and_on_the_list.

[25] A. C.-C. Yao. Protocols for secure computations (extended abstract). In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pages 160–164, 1982.

[26] A. C.-C. Yao. How to generate and exchange secrets (extended abstract). In *Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science*, pages 162–167, 1986.

Two

Protection and Retrieval of Encrypted Multimedia Content: When Cryptography Meets Signal Processing

This chapter, excluding the last section, has been published as "Protection and Retrieval of Encrypted Multimedia Content: When Cryptography Meets Signal Processing", by Z. Erkin, A. Piva, S. Katzenbeisser, R. L. Lagendijk, J. Shokrollahi, G. Neven and M. Barni in the *Eurasip Journal on Information Security*, 2007.

Abstract

The processing and encryption of multimedia content are generally considered sequential and independent operations. In certain multimedia content processing scenarios, it is however, desirable to carry out processing directly on encrypted signals. The field of secure signal processing poses significant challenges for both signal processing and cryptography research; only few ready to go fully integrated solutions are available. This paper first concisely summarizes cryptographic primitives used in existing solutions to processing of encrypted signals, and discusses implications of the security requirements on these solutions. The paper then continues to describe two domains in which secure signal processing has been taken up as a challenge, namely analysis and retrieval of multimedia content, and multimedia content protection. In each domain, state-of-the-art algorithms are described. Finally, the paper discusses the challenges and open issues in the field of secure signal processing.

2.1 Introduction

In the past few years, the processing of encrypted signals has emerged as a new and challenging research field. The combination of cryptographic techniques and signal processing is not new. So far, encryption was always considered as an add-on after signal manipulations had taken place (see Figure 2.1). For instance, when encrypting compressed multimedia signals such as audio, images, and video, first the multimedia signals were compressed using state-of-the-art compression techniques, and next encryption of the compressed bit stream using a symmetric cryptosystem took place. Consequently, the bit stream must be decrypted before the multimedia signal can be decompressed. An example of this approach is JPSEC, the extension of the JPEG2000 image compression standard. This standard adds selective encryption to JPEG2000 bit streams in order to provide secure scalable streaming and secure transcoding [45].

In several application scenarios, however, it is desirable to carry out signal processing operations directly on encrypted signals. Such an approach is called *secure signal processing*, *encrypted signal processing*, or *signal processing in the encrypted domain*. For instance, given an encrypted image, can we calculate the mean value of the encrypted image pixels? On the one hand, the relevance of carrying out such signal manipulations – i.e. the algorithm – directly on encrypted signals is entirely dependent on the security requirements of the application scenario under consideration. On the other hand, the particular implementation of the signal processing algorithm will be determined strongly by the possibilities and impossibilities of the cryptosystem employed. Finally, it is very likely that new requirements for cryptosystems will emerge

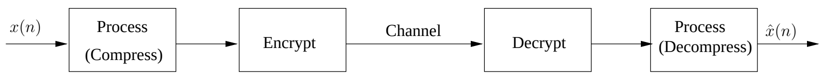


Figure 2.1: Separate processing and encryption of signals.

from secure signal processing operations and applications. Hence, secure signal processing poses a joint challenge for both the signal processing and the cryptographic community.

The security requirements of signal processing in encrypted domains depends strongly on the considered application. In this survey paper we take an application-oriented view on secure signal processing and give an overview of published applications in which the secure processing of signal amplitudes plays an important role. In each application, we show how signal processing algorithms and cryptosystems are brought together. It is not the purpose of the paper to describe either the signal processing algorithms or the cryptosystems in great detail, but rather focus on possibilities, impossibilities, and open issues in combining the two. The paper includes many references to literature that contains more elaborate signal processing algorithms and cryptosystem solutions for the given application scenario. It is also crucial to state that the scenarios in this survey can be implemented more efficiently by using trusted third entities. However, it is not always easy to find trusted entities —with high computational power, and even if one is found, it is not certain that it can be applicable in these scenarios. Therefore, the trusted entities either do not exist or have little role in discussed scenarios in this paper.

In this paper we will survey applications that directly manipulate encrypted signals. When scanning the literature on secure signal processing, it becomes immediately clear that there are currently two categories under which the secure signal processing applications and research can be roughly classified, namely content retrieval and content protection. Although the security objectives of these application categories differ quite strongly, similar signal processing considerations and cryptographic approaches show up. The common cryptographic primitives are addressed in Section 2.2. This section also discusses the need for clearly identifying the security requirements of the signal processing operations in a given scenario. As we will see, many of the approaches for secure signal processing are based on homomorphic encryption, zero-knowledge proof protocols, commitment schemes, and multiparty computation. We will also show that there is ample room for alternative approaches to secure signal processing towards the end of Section 2.2. Section 2.3 surveys secure signal processing approaches that can be classified as “content retrieval”, among them secure clustering and recommendation problems. Section 2.4 discusses problems of content protection, such as secure watermark embedding and detection. Finally, Section 2.5 concludes this chapter with the formal problem definition of this thesis.

2.2 Encryption Meets Signal Processing

2.2.1 Introduction

The capability to manipulate signals in their encrypted form is largely thanks to two assumptions on the encryption strategies used in all applications discussed. In the first place, encryption is carried out independently on individual signal samples. As a consequence, individual signal samples can be identified in the encrypted version of the signal, allowing for processing of encrypted signals on a sample-by-sample basis.

If we represent a one-dimensional (e.g. audio) signal $\mathbf{X}$ that consists of M samples as

$$\mathbf{X} = [x_1, x_2, x_3, \ldots, x_{M-1}, x_M]^T, \tag{2.1}$$

where x_i is the amplitude of the i^{th} signal sample, then the encrypted version of $\mathbf{X}$ using key k is given as

$$E_k(\mathbf{X}) = [E_k(x_1), E_k(x_2), E_k(x_3), \ldots, E_k(x_{M-1}), E_k(x_M)]^T. \tag{2.2}$$

Here the superscript "T" refers to vector transposition. Note that no explicit measures are taken to hide the temporal or spatial structure of the signal—however, the use of sophisticated encryption schemes that are *semantically secure* (as the one in [58]) achieves this property automatically.

Secondly, only *public* key cryptosystems are used that have particular *homomorphic* properties. The homomorphic property that these public key cryptographic system provide, will be concisely discussed in Section 2.2.2. In simple terms, the homomorphic property allows for carrying out additions or multiplications on signal amplitudes in the encrypted domain. Public key systems are based on the intractability of some computationally complex problems, such as

- the discrete logarithm in finite field with a large (prime) number of elements (e.g., ElGamal cryptosystem [35]),
- factoring large composite numbers (e.g., RSA cryptosystem [69]),
- deciding if a number is an n^{th} power in $\mathbb{Z}_{n^2}$ for large enough composite n (e.g., Paillier cryptosystem [58]).

It is important to realize that public key cryptographic systems operate on very large algebraic structures. This means that signal amplitudes x_i that were originally represented in 8 to 16 bits, will require at least 512 or 1024 bits per signal sample in their encrypted form $E_k(x_i)$. This data expansion is usually not emphasized in literature but this may be an important hurdle for practical applicability of secure signal processing solutions. In some cases however, several signal samples can be packed into one encrypted value in order to reduce the size of the whole encrypted signal by a linear factor [60].

A characteristic of signal amplitudes x_i is that they are usually within a limited range of values, due to the 8 to 16 bits amplitude representation format of sampled signals. If a deterministic encryption scheme would be used, each signal amplitude would always give rise to the same encrypted value, making it easy for an adversary to infer information about the signal. Consequently, probabilistic encryption has to be used, where each encryption uses a randomization or blinding factor such that even if two signal samples x_i and x_j have the same amplitude, their encrypted values $E_{pk}[x_i]$ and $E_{pk}[x_j]$ will be different. Here pk refers to the public key used upon encrypting the signal amplitudes. Public key cryptosystems are constructed such that the decryption uses only the private key sk, and that decryption does not need the value of the randomization factor used in the encryption phase. All encryption schemes that achieve the desired strong notion of *semantic security* are necessarily probabilistic.

Cryptosystems operate on (positive) integer values on finite algebraic structures. Although sampled signal amplitudes are normally represented in 8 to 16 bit (integer) values when they are stored, played, or displayed, intermediate signal processing operations often involve non-integer signal amplitudes. Work-arounds for non-integer signal amplitudes may involve scaling signal amplitudes with constant factors (say factors of 10 to 1000), but the unavoidable successive operations of rounding (quantization) and normalization by division pose significant challenges for being carried out on encrypted signal amplitudes.

In Section 2.2.2 we first discuss four important cryptographic primitives that are used in many secure signal processing applications, namely homomorphic encryption, zero knowledge proof protocols, commitment schemes, and secure multiparty computation. In Section 2.2.3 we then consider the importance of scrutinizing the security requirements of the signal processing application. It is meaningless to speak about secure signal processing in a particular application if the security requirements are not specified. The security requirements as such will also determine the possibility or impossibility of applying the cryptographic primitives. As we will illustrate by examples—and also in more detail in the following sections—some application scenarios simply cannot be made secure because of the inherent information leakage by the signal processing operation, because of the limitations of the cryptographic primitives to be used, or because of constraints on the number of interactions between parties involved. Finally, in Section 2.2.4 we briefly discuss the combination of signal encryption and compression using an approach quite different from the ones discussed in Sections 3 and 4, namely by exploiting the concept of coding with side information. We discuss this approach here to emphasize that although many of the currently existing application scenarios are built on the four cryptographic primitives discussed in Section 2.2.2, there is ample room for entirely different approaches to secure signal processing.

2.2.2 Cryptographic Primitives

Homomorphic Cryptosystems

Many signal processing operations are linear in nature. Linearity implies that multiplying and adding signal amplitudes are important operations. At the heart of many signal processing operations, such as linear filters and correlation evaluations, is the calculation of the inner product between two signals $\mathbf{X}$ and $\mathbf{Y}$. If both signals (or segments of the signals) contain M samples, then the inner product is defined as:

$$< \mathbf{X}, \mathbf{Y} >= \mathbf{X}^T \mathbf{Y} \quad = \quad [x_1, x_2, \ldots, x_M] \cdot \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \sum_{i=1}^{M} x_i y_i. \qquad (2.3)$$

This operation can be carried out directly on an encrypted signal $\mathbf{X}$ and plain text signal $\mathbf{Y}$ if the encryption system used has the additive homomorphic property, as we will discuss next.

Formally, a (public key) encryption system $E_{pk}(\cdot)$ and its decryption $D_{sk}(\cdot)$ are homomorphic if those two functions are maps between the message group with an operation $f_1(\cdot)$ and the encrypted group with an operation $f_2(\cdot)$, such that if x and y are taken from the message space of the encryption scheme, we have:

$$f_1(x, y) \quad = \quad D_{sk}(f_2(E_{pk}(x), E_{pk}(y))). \tag{2.4}$$

For secure signal processing, multiplicative and additive homomorphisms are important. Table 2.1 gives an overview of encryption systems with additive or multiplicative homomorphism. Note that those homomorphic operations are applied to a modular domain (i.e., either in a finite field or in a ring $\mathbb{Z}_n$)—thus, both addition and multiplication are taken modulo some fixed value. For signal processing applications, which usually require integer addition and multiplication, it is thus essential to choose the message space of the encryption scheme large enough so that overflows due to modular arithmetic are avoided when operations on encrypted data are performed.

Another important consideration is the representation of the individual signal samples. As encryption schemes usually operate in finite modular domains (and all messages to be encrypted must be represented in this domain), a mapping is required which quantizes real-valued signal amplitudes and translates the signal samples of $\mathbf{X}$ into a vector of modular numbers. In addition to the requirement that the computations must not overflow, special care must be taken to represent negative samples in a way which is compatible with the homomorphic operation offered by the cryptosystem. For the latter problem, depending on the algebraic structure of the cipher, one may either encode the negative value $-x$ by the modular inverse x^{-1} in the underlying algebra of the message space or by avoiding negative numbers entirely by using a constant additive shift.

In the context of the above inner product example, we require an additively homomorphic scheme (see Table 2.1). Hence, f_1 is the addition, and f_2 is a multiplication:

$$x + y \quad = \quad D_{sk}(E_{pk}(x) \cdot E_{pk}(y)), \tag{2.5}$$

or equivalently:

$$E_{pk}(x + y) \quad = \quad E_{pk}(x) \cdot E_{pk}(y). \tag{2.6}$$

Note that the latter equation also implies that

$$E_{pk}(c \cdot x) \quad = \quad (E_{pk}(x))^c \tag{2.7}$$

for every integer constant c. Thus, every additively homomorphic cryptosystem also allows to multiply an encrypted value with a constant available or known as clear text.

The Paillier cryptosystem [58] provides the required homomorphism, if both addition and multiplication are considered as modular. The encryption of a message m under a Paillier cryptosystem is defined as

$$E_{pk}(m) \quad = \quad g^m r^n \mod n^2, \tag{2.8}$$

where $n = pq$, p and q are large prime number, $g \in \mathbb{Z}^*_{n^2}$ is a generator whose order is a multiple of n, and $r \in \mathbb{Z}^*_n$ is a random number (blinding factor). We then easily see that

$$\begin{aligned} E_{pk}(x)E_{pk}(y) &= (g^x r_x^n)(g^y r_y^n) \mod n^2 \\ &= g^{x+y}(r_x r_y)^n \mod n^2 \\ &= E_{pk}(x+y). \end{aligned} \tag{2.9}$$

Applying the additive homomorphic property of the Paillier encryption system, we can evaluate Eq. (2.3) under the assumption that $\mathbf{X}$ is an encrypted signal and $\mathbf{Y}$ is a plain text signal:

$$E_{pk} < \mathbf{X}, \mathbf{Y} > \; = \; E_{pk}\left(\sum_{i=1}^{M} x_i y_i\right) = \prod_{i=1}^{M} E_{pk}(x_i y_i) = \prod_{i=1}^{M} E_{pk}(x_i)^{y_i} \tag{2.10}$$

Here we implicitly assume that x_i, y_i are represented as integers in the message space of the Paillier cryptosystem, i.e. $x_i, y_i \in \mathbb{Z}_n$. Equation (2.10) essentially shows that it is possible to compute an inner product directly in case one of the two vectors is encrypted. One takes the encrypted samples $E_{pk}(x_i)$, raises them to the power of y_i and multiplies all obtained values. Obviously, the resulting number itself is also in encrypted form. To carry out further useful signal processing operations on the encrypted result, for instance to compare it to a threshold, another cryptographic primitive is needed, namely zero knowledge proof protocols, which is discussed in the next section.

In the paper we focus mainly on public-key encryption schemes, as almost all homomorphic encryption schemes belong to this family. The notable exception is the one-time pad (and derived stream ciphers), where messages taken from a finite group are blinded by a sequence of uniformly random group elements. Despite its computationally efficient encryption and decryption processes, the application of a one-time pad usually raises serious problems with regard to key distribution and management. Nevertheless, it may be used to temporarily blind intermediate values in larger communication protocols. Finally, it should be noted that some recent work in cryptography (like searchable encryption [11] and order preserving encryption [4]) may also yield alternative ways for the encryption of signal samples. However, these approaches have not yet been studied in the context of media encryption.

To conclude this section, we observe that directly computing the inner product of *two* encrypted signals is not possible since this would require a cryptographic system that has both multiplicative and additive (i.e., algebraic) homomorphism. Recent proposals in that direction like [27, 28] were later proven to be insecure [77, 17]. Therefore, no *provably secure* cryptographic system with these properties is known to date. The construction of an algebraic privacy homomorphism remains an open problem. Readers can refer to [32] for more details on homomorphic cryptosystems.

Zero-Knowledge Proof Protocols

Zero-knowledge protocols are used to prove a certain statement or condition to a verifier, without revealing any "knowledge" to the verifier except the fact that the assertion is valid [38]. As a simple example, consider the case where the prover Peggy claims

Table 2.1: Some (probabilistic) encryption systems and their homomorphisms.

Encryption system	$f_1(.,.)$	$f_2(.,.)$
Multiplicatively Homomorphic El-Gamal [35]	multiplication	multiplication
Additively Homomorphic El-Gamal [72]	addition	multiplication
Goldwasser-Micali [40]	XOR	multiplication
Benaloh [10]	addition	multiplication
Naccache-Stern [56]	addition	multiplication
Okamoto-Uchiyama [57]	addition	multiplication
Paillier [58]	addition	multiplication
Damgård-Jurik [26]	addition	multiplication

to have a way of factorizing large numbers. The verifier Victor will send her a large number and Peggy will send back the factors. Successful factorization of several large integers will decrease Victor's doubt in the truth of Peggy's claim. At the same time Victor will learn "no knowledge of the actual factorization method".

Although simple, the example shows an important property of zero-knowledge protocol proofs, namely that they are interactive in nature. The interaction should be such that with increasing number of "rounds", the probability of an adversary to successfully prove an invalid claim decreases significantly. On the other hand, non-interactive protocols (based on the random oracle model) also do exist. A formal definition of interactive and non-interactive proof systems, such as zero-knowledge protocols, falls outside the scope of this paper, but can be found for instance in [38].

As an example for a commonly used zero-knowledge proof, consider the proof of knowing the discrete logarithm x of an element y to the base g in a finite field [71]. Having knowledge of discrete logarithm x is of interest in some applications since if

$$y \quad = \quad g^x \mod p, \tag{2.11}$$

then given p (a large prime number), g and y the calculation of the logarithm x is computationally infeasible. If Peggy (the prover) claims she knows the answer (i.e., the value of x), she can convince Victor (the verifier) of this knowledge without revealing the value of x by the following zero-knowledge protocol. Peggy picks a random number $r \in \mathbb{Z}_p$, and computes $t = g^r \mod p$. She then sends t to Victor. He picks a random challenge $c \in \mathbb{Z}_p$ and sends this to Peggy. She computes $s = r - cx \mod p$ and sends this to Victor. He accepts Peggy's knowledge of x if $g^s y^c = t$, since if Peggy indeed used the correct logarithm x in calculating the value of s, we have

$$g^s y^c \mod p \quad = \quad g^{r-cx}(g^x)^c \mod p = g^r = t \mod p. \tag{2.12}$$

In literature, many different zero-knowledge proofs exist. We mention a number of them that are frequently used in secure signal processing:

- proof that an encrypted number is non-negative [53];
- proof that shows that an encrypted number lies in a certain interval [12];

- proof that the prover knows the plaintext x corresponding to the encryption $E(x)$ [33];
- proofs that committed values (see Section 2.2.2) satisfy certain algebraic relations [13].

In zero-knowledge protocols, it is sometimes necessary for the prover to commit to a particular integer or bit value. Commitment schemes are discussed in the next section.

Commitment Schemes

An integer or bit commitment scheme is a method that allows Alice to commit to a value while keeping it hidden from Bob, and while also preserving Alice's ability to reveal the committed value later to Bob. A useful way to visualize a commitment scheme is to think of Alice as putting the value in a locked box, and giving the box to Bob. The value in the box is hidden from Bob, who cannot open the lock (without the help of Alice), but since Bob has the box, the value inside cannot be changed by Alice; hence, Alice is "committed" to this value. At a later stage, Alice can "open" the box and reveal its content to Bob.

Commitment schemes can be built in a variety of ways. As an example, we review a well-known commitment scheme due to Pedersen [61]. We fix two large primes p and q such that $q|(p-1)$ and a generator g of the subgroup of order q of $\mathbb{Z}_p^*$. Furthermore, we set $h = g^a \mod p$ for some random secret a. The values p, q, g and h are the public parameters of the commitment scheme. To commit to a value m, Alice chooses a random value $r \in \mathbb{Z}_q$ and computes the commitment $c = g^m h^r \mod p$. To open the commitment, Alice sends m and r to Bob, who verifies that the commitment c received previously indeed satisfies $c = g^m h^r \mod p$. The scheme is hiding due to the random blinding factor r; furthermore, it is binding unless Alice is able to compute discrete logarithms.

For use in signal processing applications, commitment schemes that are additively homomorphic are of specific importance. As with homomorphic public key encryption schemes, knowledge of two commitments allows one to compute—without opening—a commitment of the sum of the two committed values. For example, the above mentioned Pedersen commitment satisfies this property: given two commitments $c_1 = g^{m_1} h^{r_1} \mod p$ and $c_2 = g^{m_2} h^{r_2} \mod p$ of the numbers m_1 and m_2, a commitment $c = g^{m_1+m_2} h^{r_1+r_2} \mod p$ of $m_1 + m_2$ can be computed by multiplying the commitments: $c = c_1 c_2 \mod p$. Note that the commitment c can be opened by providing the values $m_1 + m_2$ and $r_1 + r_2$. Again, the homomorphic property only supports additions. However, there are situations where it is not possible to prove the relation by mere additive homomorphism as in proving that a committed value is the square of the value of another commitment. In such circumstances, zero-knowledge proofs can be used. In this case, the party which possesses the opening information of the commitments computes a commitment of the desired result, hands it to the other party and proves in zero-knowledge that the commitment was actually computed in the correct manner. Among others, such zero-knowledge proofs exist for all polynomial relations between committed values [13].

Secure Multiparty Computation

The goal of SMC is to evaluate a public function $f(x^{(1)}, x^{(2)}, \ldots, x^{(m)})$ based on the secret inputs $x^{(i)}, i = 1, 2, \ldots, m$ of m users, such that the users learn nothing except their own input and the final result. A simple example, called Yao's Millionaire's Problem, is the comparison of two (secret) numbers in order to determine if $x^{(1)} > x^{(2)}$. In this case the parties involved will only learn if their number is the largest, but nothing more than that.

There is a large body of literature on secure multiparty computation; for example, it is known [79] that any (computable) function can be evaluated securely in the multiparty setting by using a general circuit-based construction. However, the general constructions usually require a large number of interactive rounds and a huge communication complexity. For practical applications in the field of distributed voting, private bidding and auctions, and private information retrieval, dedicated lightweight multiparty protocols have been developed. An example relevant to signal processing application is the multiparty computation known as Bitrep which finds the encryption of each bit in the binary representation of a number whose encryption under an additive homomorphic cryptosystem is given [73]. We refer the reader to [39] for an extensive summary of secure multiparty computations and [18] for a brief introduction.

2.2.3 Importance of Security Requirements

Although the cryptographic primitives that we discussed in the previous section are useful for building secure signal processing solutions, it is important to realize that in each application the security requirements have to be made explicit right from the start. Without wishing to turn to formal definition, we choose to motivate the importance of what to expect from secure signal processing with three simple yet illustrative two-party computation examples.

The first simple example is the encryption of a (say audio) signal $\mathbf{X}$ that contains M samples. Due to the sample-by-sample encryption strategy as shown in Eq. (2.2), the encrypted signal $E_{pk}(\mathbf{X})$ will also contain M encrypted values. Hence, the *size* M of the plain text signal cannot be hidden by the approaches followed in secure signal processing surveyed in this paper.

In the second example, we consider the linear filtering of the signal $\mathbf{X}$. In a (FIR) linear filter, the relation between the input signal amplitudes $\mathbf{X}$ and output signal amplitudes $\mathbf{Y}$ is entirely determined by the impulse response $(h_0, h_1, \ldots, h_r)$ through the following convolution equation:

$$y_i \quad = \quad h_0 x_i + h_1 x_{i-1} + \ldots + h_r x_{i-r} = \sum_{k=0}^{r} h_k x_{i-k}. \tag{2.13}$$

Let us assume that we wish to compute this convolution in a secure way. The first party, Alice, has the signal $\mathbf{X}$ and the second party, Bob, has the impulse response $(h_0, h_1, \ldots, h_r)$. Alice wishes to carry out the convolution (2.13) using Bob's linear filter. However, both Bob and Alice wish to keep secret their data, i.e., the impulse response and the input signal, respectively. Three different setups can now be envisioned.

- Alice encrypts the signal $\mathbf{X}$ under an additive homomorphic cryptosystem and sends the encrypted signal to Bob. Bob then evaluates the convolution (2.13) on the encrypted signal as follows:

$$\begin{aligned} E_{pk_A}(y_i) &= E_{pk_A}\left(\sum_{k=0}^{r} h_k x_{i-k}\right) \\ &= \prod_{k=0}^{r} E_{pk_A}(h_k x_{i-k}) = \prod_{k=0}^{r} E_{pk_A}(x_{i-k})^{h_k}. \end{aligned} \tag{2.14}$$

  Notice that the additive homomorphic property is used in the above equation and that indeed individually encrypted signal samples should be available to Bob. Also notice that the above evaluation is only possible if both $\mathbf{X}$ and $(h_0, h_1, \ldots, h_r)$ are integer-valued, which is actually quite unlikely in practice. After computing Eq. (2.14), Bob sends the result back to Alice who decrypts the signal using her private key to obtain the result $\mathbf{Y}$. In this setup Bob does not learn the output signal $\mathbf{Y}$.

- Bob encrypts his impulse response $(h_0, h_1, \ldots, h_r)$ under a homomorphic cryptosystem and sends the result to Alice. Alice then evaluates the convolution (2.13) using the encrypted impulse response as follows:

$$\begin{aligned} E_{pk_B}(y_i) &= E_{pk_B}\left(\sum_{k=0}^{r} h_k x_{i-k}\right) \\ &= \prod_{k=0}^{r} E_{pk_B}(h_k x_{i-k}) = \prod_{k=0}^{r} E_{pk_B}(h_k)^{x_{i-k}}. \end{aligned} \tag{2.15}$$

  Alice then sends the result to Bob, who decrypts to obtain the output signal $\mathbf{Y}$. In this solution Bob learns the output signal $\mathbf{Y}$.

- Alice and Bob engage in a formal multiparty protocol, where the function to be evaluated $f(x_1, x_2, \ldots, x_M, h_0, h_1, \ldots, h_r)$ is the convolution equation, Alice holds the signal values x_i and Bob the impulse response h_i as secret inputs. Both parties will learn the resulting output signal $\mathbf{Y}$.

Unfortunately, none of the above three solutions really provides a solution to the secure computation of a convolution due to inherent algorithm properties. For instance, in the first setup, Alice could send Bob a signal that consists of all-zero values and a single "one" value (a so-called "impulse signal"). After decrypting the result $E_{pk_A}(y_i)$ that she obtains from Bob, it is easy to see that $\mathbf{Y}$ is equal to $(h_0, h_1, \ldots, h_r)$, hence Bob's impulse response is subsequently known to Alice. Similar attacks can be formulated for the other two cases. In fact, even for an arbitrary input both parties can learn the other's input by a well-known signal processing procedure known as "deconvolution". In conclusion, although in some cases there may be a need for the secure evaluation of convolutions, the inherent properties of the algorithm make secure computing in a two-party scenario meaningless. (Nevertheless, the protocols have value if used as building blocks in a large application where the output signal $\mathbf{Y}$ is not revealed to the attacker.)

The third and final example is to threshold a signal's (weighted) mean value in a secure way. The (secure) mean value computation is equivalent to the (secure) computation of the inner product Eq. (2.3), with $\mathbf{X}$ the input signal and $\mathbf{Y}$ the weights that define how the mean value is calculated. In the most simple case, we have $y_i = 1$ for all i, but other definitions are quite common. Let use assume that Alice wishes Bob to determine if the signal's mean value is "critical", for instance above a certain threshold value T_c, without revealing $\mathbf{X}$ to Bob. Bob on the other hand does not want to reveal his expert knowledge, namely the weights $\mathbf{Y}$ and the threshold T_c. Two possible solutions to this secure decision problem are the following.

- Use secure multiparty computation, where the function $f(\cdot)$ is a combination of the inner product and threshold comparison. Both parties will only learn if the mean value is critical or not.
- Alice sends Bob the signal $\mathbf{X}$ under additively homomorphic encryption. Bob securely evaluates the inner product using Eq. (2.10). After encrypting T_c using Alice's public key, Bob computes the (encrypted version of the) difference between the computed mean and threshold T_c. Bob sends the result to Alice, who decrypts the result using her secret key and checks if the value is larger or smaller than zero.

Although the operations performed are similar to the second example, in this example the processing is secure since Bob learns little about Alice's signal and Alice will learn little about the Bob's expert knowledge. In fact, in the first implementation the entire signal processing operation is ultimately condensed into a single bit of information; the second implementation leaks more information, namely the distance between the correlation value from the threshold. In both cases, the result represents a high information abstraction level, which is insufficient for launching successful signal processing-based attacks. In contrast, in the example based on Eq. (2.13) the signal processing operation led to an enormous amount of information—the entire output signal—to be available to either parties, making signal processing-based attacks quite easy.

As we will see in Sections 2.3 and 2.4, many of the two-party secure signal processing problems eventually include an information condensation step, such as (in the most extreme case) a binary decision. We postulate that for two-party linear signal processing operations in which the amount of plain text information after processing is in the same order of magnitude as before processing, no secure solutions exist purely based on the cryptographic primitives discussed in the previous section, due to inherent properties of the signal processing problems and the related application scenario. For that reason, entirely other approaches to secure signal processing are also of interest. Although few results can be found in literature on approaches not using homomorphic encryption, zero-knowledge proofs, and multiparty computation protocols, the approach discussed in the next section may well show a possible direction for future developments.

2.2.4 Compression of Encrypted Signals

When transmitting signals that contain redundancy over an insecure and bandwidth constrained channel, it is customary to first compress and then encrypt the signal. Using the principles of coding with side information, it is however also possible to interchange the order of (lossless) compression and encryption, i.e. to compress *encrypted* signals [44]. The concept of swapping the order of compression and encryption is illustrated in Figure 2.2. A signal from the message source is first encrypted and then compressed. The compressor does *not* have access to the secret key used in the encryption. At the decoder, decompression and decryption are performed jointly. From classical information theory, it would seem that only minimal gain could be obtained as the encrypted signal has maximal entropy, i.e. no redundancy is left after encryption. However, the decoder can use the cryptographic key to *decode and decrypt* the compressed and encrypted bit stream. This brings opportunities for efficient compression of encrypted signals based on principle of coding with side information. In [44], it was shown that neither compression performance nor security need to be negatively impacted under some reasonable conditions.

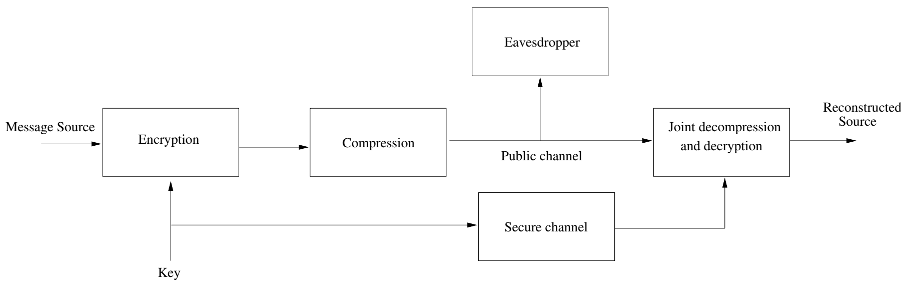


Figure 2.2: Compression of an encrypted signal, from [44].

In source coding with side information, the signal $\mathbf{X}$ is coded under the assumption that the decoder—but not the encoder—has statistically dependent information $\mathbf{Y}$, called the side information, available. In conventional coding scenarios, the encoder would code the difference signal $\mathbf{X} - \mathbf{Y}$ in some efficient way, but in source coding with side information this is impossible since we assume that $\mathbf{Y}$ is only known at the decoder. In the Slepian-Wolf coding theory [74], the crucial observation is that the side information $\mathbf{Y}$ is regarded as a degraded version of $\mathbf{X}$. The degradations are modeled as "noise" on the "virtual channel" between $\mathbf{X}$ and $\mathbf{Y}$. The signal $\mathbf{X}$ can then be recovered from $\mathbf{Y}$ by the decoder if sufficient error correcting information is transmitted over the channel. The required bit rate and amount of entropy are related as $R \geq H(\mathbf{X}|\mathbf{Y})$. This shows that, at least theoretically, there is no loss in compression efficiency since the lower bound $H(\mathbf{X}|\mathbf{Y})$ is identical to the scenario in which $\mathbf{Y}$ is available at the encoder. Extension of the Slepian-Wolf theory exists for lossy source coding [67]. In all practical cases of interests, the information bits that are transmitted over the channel are parity bits or syndromes of channel coding methods such as Hamming, Turbo or LDPC codes.

In the scheme depicted in Figure 2.2 we have a similar scenario as in the above source coding with side information case. If we consider the encrypted signal $E_k(\mathbf{X})$ at the input of the encoder, then we see that the decoder has the key k available, representing the “statistically dependent side information”. Hence, according to the Slepian-Wolf viewpoint, the encrypted signal $E_k(\mathbf{X})$ can be compressed to a rate that is the same as if the key k would be available during the source encoding process, that is, $R \geq H(E_k(\mathbf{X})|k) = H(\mathbf{X})$. This clearly says that the (lossless) coding of the encrypted signal $E_k(\mathbf{X})$ should be possible with the same efficiency as the (lossless) coding of $\mathbf{X}$. Hence, using the side information key k, the decoder can recover first $E_k(\mathbf{X})$ from the compressed channel bit stream and subsequently decode $E_k(\mathbf{X})$ into $\mathbf{X}$.

A simple implementation of the above concept for a binary signal $\mathbf{X}$ uses a pseudo randomly generated key. The key k is in this case a binary signal $\mathbf{K}$ of the same dimension M as the signal $\mathbf{X}$. The encrypted signal is computed as follows:

$$\begin{aligned} E_k(\mathbf{X}) &= \mathbf{X} \oplus \mathbf{K}, \\ E_k(x_i) &= x_i \oplus k_i \quad i = 1, 2, \ldots, M. \end{aligned} \tag{2.16}$$

The encrypted signal $E_k(\mathbf{X})$ is now input to a channel coding strategy, for instance a Hamming coding. The strength of the Hamming code is dependent on the dependency between $E_k(\mathbf{X})$ and the side information $\mathbf{K}$ at the decoder. This strength obviously depends solely on the properties of the original signal $\mathbf{X}$. This does, however, require the message source to inform the source encoder about the entropy $H(\mathbf{X})$, which represents a small leak of information. The encoder calculates parity check bits over binary vectors of some length L created by concatenating L bits of the encrypted signal $E_k(\mathbf{X})$, and sends *only these parity check bits* to the receiver.

The decoder recovers the encrypted signal by first appending to $\mathbf{K}$ the parity check bits, and then error correcting the resulting bit pattern. The success of this error correction step depends on the strength of the Hamming code, but as mentioned, this strength has been chosen sufficiently with regards to the “errors” in $\mathbf{K}$ on the decoding side. Notice that in this particular setup the “errors” represent the bits of the original signal $\mathbf{X}$. If the error correction step is successful, the decoder obtains $E_k(\mathbf{X})$, from which the decryption can straightforwardly take place:

$$\begin{aligned} \mathbf{X} &= E_k(\mathbf{X}) \oplus \mathbf{K}, \\ x_i &= E_k(x_i) \oplus k_i \quad i = 1, 2, \ldots, M. \end{aligned} \tag{2.17}$$

The above example is too simple for any practical scenario for a number of reasons. In the first place, it uses only binary data, for instance bit planes. More efficient coding can be obtained if the dependencies between bit planes are considered. This effectively requires an extension of the bit plane coding and encryption approach to coding and encryption of symbol values. Secondly, the decoder lacks a model of the dependencies in $\mathbf{X}$. Soft decoders for Turbo or LDPC codes can exploit such message source models, yielding improved performance. Finally, the coding strategy is lossless. For most continuous or multi-level message sources, such as audio, images, and video, lossy compression is desirable.

2.3 Analysis and Retrieval of Content

In the today's society, huge quantities of personal data are gathered from people and stored in databases for various purposes ranging from medical researches to online personalized applications. Sometimes providers of these services may want to combine their data for research purposes. A classical example is the one where two medical institutions wish to perform joint research on the union of their patients data. Privacy issues are important in this scenario because the institutions need to preserve their private data during their cooperation. Lindell and Pinkas [52], and Agrawal and Srikant [5] proposed the notion of privacy preserving data mining, meaning the possibility to perform data analysis from distributed database, under some privacy constraints. Privacy preserving data mining [63, 46, 19, 76] deals with mutual untrusted parties that on the one hand wish to cooperate to achieve a common goal but, on the other hand, are not willing to disclose their knowledge to each other.

There are several solutions that cope with exact matching of data in a secure way. However, it is more common in signal processing to perform inexact matching, i.e. learning the distance between two signal values, rather than exact matching. Consider two signal values x_1 and x_2. Computing the distance between them or checking if the distance is within a threshold is important:

$$|x_1 - x_2| < \epsilon. \tag{2.18}$$

This comparison or *fuzzy matching* can be used in a variety of ways in signal processing. One example is quantizing data which is of crucial importance for multimedia compression schemes. However, considering that these signal values are encrypted —thus the ordering between them is totally destroyed, there is not any efficient way known to *fuzzy* compare two values.

In the following sections, we give a summary of techniques that focus on extracting some information from protected datasets. Selected studies mostly use homomorphic encryption, zero-knowledge proofs and sometimes multiparty computations. As we will see, most solutions still require substantial improvements in communication and computation efficiency in order to make them applicable in practice. Therefore, the last section addresses a different approach that uses other means of preserving privacy to show that further research on combining signal processing and cryptography may result in new approaches rather than using encryption schemes and protocols.

2.3.1 Clustering

Clustering is a well-studied combinatorial problem in data mining [43]. It deals with finding a structure in a collection of unlabeled data. One of the basic algorithms of clustering is the K-means algorithm that partitions a data set into K clusters with a minimum error. We review the K-means algorithm and its necessary computations such as distance computation and finding the cluster centroid, and show that cryptographic protocols can be used to provide user's privacy in clustering for certain scenarios.

K-means Clustering Algorithm

The K-means clustering algorithm partitions a dataset DB of "objects" such as signal values or features thereof, into K disjoint subsets, called clusters. Each cluster is represented by its center which is the centroid of all objects in that subset.

Algorithm 1 The K-means clustering algorithm

```
1: Select $K$ random objects representing the $K$ initial centroid of the clusters.
2: Assign each object to the cluster with the nearest centroid.
3: Recalculate the centroids for each cluster.
4: Repeat step 2 and 3 until centroids do not change or a certain threshold achieved.
```

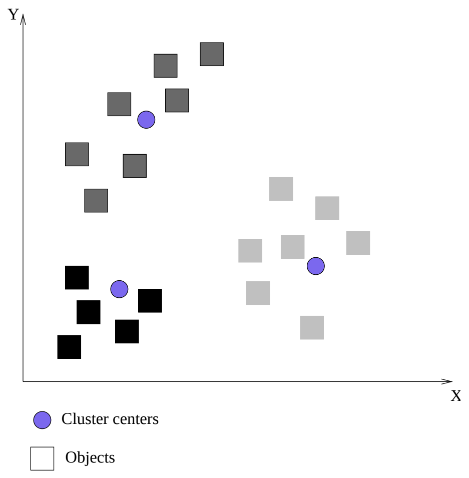


Figure 2.3: Clustered dataset. Each object is a point in the 2-dimensional space. K-means clustering algorithm assigns each object to the cluster with the smallest distance.

As shown in Algorithm 1, the K-means algorithm is an iterative procedure that refines the cluster centroids until a predefined condition is reached. The algorithm first chooses K random points as the cluster centroids in the dataset DB and assigns the objects to the closest cluster centroid. Then, the cluster centroid is re-computed with recently assigned objects. When the iterative procedure reaches the termination condition, each data object is assigned to the closest cluster (Figure 2.3). Thus to carry out the K-means algorithm, the following quantities needs to be computed:

- the cluster centroid, or the mean of the data objects in that cluster,

- the distance between an object and the cluster centroid,
- the termination condition which is a distance measurement compared to a threshold.

In the following section we describe a secure protocol that carries out secure K-means algorithm on protected data objects.

Secure K-means Clustering Algorithm

Consider the scenario in which Alice and Bob want to apply the K-means algorithm on their joint datasets as shown in Figure 2.4, but at the same time they want to keep their own dataset private. Jagannathan *et al.* proposed a solution for this scenario in [42].

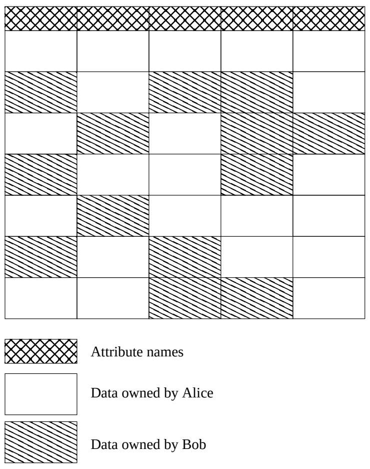


Figure 2.4: Shared dataset on which K-means algorithm is run.

In the proposed method, both Alice and Bob get the final output but the values computed in the intermediate steps are unknown to the both parties. Therefore, the intermediate values such as cluster centroids are uniformly shared between Alice and Bob in such a way that for a value x, Alice gets a random share a and Bob gets another random share b where $(a + b)$ mod $N = x$ and N is the size of the field in which all operations take place. Alice and Bob keep their private shares of the dataset secret.

The secure K-means clustering algorithm is separated into subprotocols where Alice and Bob computes the followings (Algorithm 2):

1. **Distance measurement and finding the closest cluster:** The distance between each object and cluster centroid is computed by running a secure scalar product protocol by Goethals *et al.* [36]. The closest cluster centroid is determined by running Yao's circuit evaluation protocol [78] with the shared data of Alice and Bob.

2. **New cluster centroid:** The new cluster centroid requires to determine an average computation over shared values of Alice and Bob. This function of the form $\frac{a+b}{m+n}$ can be computed by applying Yao's protocol where Alice knows a and m and Bob knows b and n.

3. **Termination condition:** The termination condition of the algorithm is computed by running the Yao's circuit evaluation protocol [78].

The squared distance between an object $\mathbf{X}_i = (x_{i,1}, \ldots, x_{i,M})$ and a cluster centroid μ_j is given by the following equation:

$$(\text{dist}(\mathbf{X}_i, \mu_j))^2 = (x_{i,1} - \mu_{j,1})^2 + (x_{i,2} - \mu_{j,2})^2 + \ldots + (x_{i,M} - \mu_{j,M})^2. \quad (2.19)$$

Considering that the clusters centroids are shared between Alice and Bob, Eq. (2.19) can be written as,

$$(\text{dist}(\mathbf{X}_i, \mu_j))^2 = (x_{i,1} - (\mu^A_{j,1} + \mu^B_{j,1}))^2 + \ldots + (x_{i,M} - (\mu^A_{j,M} + \mu^B_{j,M}))^2, \quad (2.20)$$

where μ^A_j is Alice's share and μ^B_j is Bob's share such that the jth-cluster centroid is $\mu_j = \mu^A_j + \mu^B_j$. Then, the Eq. (2.20) can be written as,

$$\begin{aligned}(\text{dist}(\mathbf{X}_i, \mu_j))^2 &= \sum_{k=1}^{M} x^2_{i,k} + \sum_{k=1}^{M} (\mu^A_{j,k})^2 + \sum_{k=1}^{M} (\mu^B_{j,k})^2 + 2\sum_{k=1}^{M} \mu^A_{j,k}\mu^B_{j,k} \\ &- 2\sum_{k=1}^{M} \mu^A_{j,k} x_{i,k} - 2\sum_{k=1}^{M} x_{i,k}\mu^B_{j,k}. \end{aligned} \quad (2.21)$$

Equation (2.21) can be computed by Alice and Bob jointly. As the first term of the equation is shared between them, Alice computes the sum of components of her share while Bob computes the rest of the components. The second term and third term can be computed by Alice and Bob individually, and the rest of the terms are computed by running a secure scalar product protocol between Alice and Bob, much similar to the evaluation of Eq. (2.3) via the secure form of Eq. (2.10). Alice first encrypts her data $E_{pk_A}(\mu^A_j) = (E_{pk_A}(\mu^A_{j,1}), \ldots, E_{pk_A}(\mu^A_{j,M}))$ and sends it to Bob who computes the scalar product of this data with his own by using the additive homomorphic property of the encryption scheme as follows:

$$E_{pk_A}(\mu^A_j)^{\mu^B_j} = (E_{pk_A}(\mu^A_{j,1})^{\mu^B_{j,1}}, \ldots, E_{pk_A}(\mu^A_{j,M})^{\mu^B_{j,M}}). \quad (2.22)$$

Then, multiplying the encrypted components gives the encrypted scalar product of Alice's and Bob's data,

$$E_{pk_A}\left(\sum_{k=1}^{M} \mu^A_{j,k}\mu^B_{j,k}\right) = \prod_{k=1}^{M} E_{pk_A}(\mu^A_{j,k})^{\mu^B_{j,k}}. \quad (2.23)$$

The computed distances between the objects and the cluster centroids can later be the input to the Yao's circuit evaluation protocol [78] in which the closest cluster centroid is determined. We refer readers to [36] and [78] for further details on this part.

Once the distances and the closest clusters to the objects are determined, each object is labeled with the nearest cluster index. At the end of each iteration it is necessary to compute the new cluster centroids. Alice computes the sum of the corresponding coordinates of all object s_j and the number of objects n_j within each of the K clusters for j, $1 \leq j \leq M$. As shown in Figure 2.4, Alice has only some of the attributes of the objects, thus she treats these missing values as zero. Bob also applies the same procedure and determines the sum of coordinates t_j and the number of objects m_j in the clusters. Given s_j, t_j, n_j and m_j, the jth component of the ith cluster is,

$$\mu_{i,j} = \frac{s_j + t_j}{n_j + m_j}. \tag{2.24}$$

Since there are only four values, this equation can be computed efficiently by using Yao's circuit evaluation protocol [78] with Alice's shares s_j and n_j and Bob's shares t_j and m_j.

In the last step of the K-means algorithm, the iteration is terminated if there is no further improvement between the previous and current cluster centroids. In order to do that, a distance is computed between the previous and current cluster centroids. This is done in the same way as computing distances between an object and a cluster centroid but in addition, this distance is compared to a threshold value ϵ. Considering that the cluster centroids are shared between Alice and Bob, the result of the computation of the squared distance of cluster centroids for the kth and $k+1$th iterations is again random shares for Alice and Bob.

$$(\text{dist}(\mu_j^{A,k+1} + \mu_j^{B,k+1}, \mu_j^{A,k} + \mu_j^{B,k}))^2 = \alpha_j + \beta_j, \tag{2.25}$$

where α and β are the shares of Alice and Bob. Alice and Bob then apply Yao's protocol on their K-length vectors $(\alpha_1, \ldots, \alpha_K)$ and $(\beta_1, \ldots, \beta_K)$ to check if $\alpha_j + \beta_j < \epsilon$ for $1 \leq j \leq K$.

Algorithm 2 Privacy preserving K-means clustering algorithm.

```
Randomly select $K$ objects from the dataset $DB$ as initial cluster centroids
Randomly share the cluster centroid between Alice and Bob
repeat
  for all object $d_k$ in dataset $DB$ do
    Run the secure closest cluster protocol
    Assign to $d_k$ to the closest cluster
  end for
  Alice and Bob computes the random shares for the new centroids of the clusters.
until cluster centroids are close to each other with an error of $\epsilon$.
```

2.3.2 Recommender Systems

Recommender services play an important role in applications like e-commerce and direct recommendations for multimedia contents. These services attempt to predict items that a user may be interested in by implementing a signal processing algorithm known as *collaborative filtering* on user preferences to find similar users that share the same taste (likes or dislikes). Once similar users are found, this information can be used in variety ways such as recommending restaurants, hotels, books, audio and video etc.

Recommender systems store user data, also known as preferences, in servers, and the collaborative filtering algorithms work on these stored preferences to generate recommendations. The amount of data collected from each user directly affects the accuracy of the predictions. There are two concerns in collecting information from the users in such systems. First, in an ordinary system there are in the order of thousands items, so that it is not realistic for the users to rate all of them. Second, users would not like to reveal too much privacy sensitive information that can be used to track them.

The first problem, also known as the sparseness problem in datasets, is addressed for collaborative filtering algorithms in [37, 68, 70]. The second problem on user privacy is of interest to this survey paper since users tend to not give more information about themselves for privacy concerns and yet they expect more accurate recommendations that fit their taste. This tradeoff between privacy and accuracy leads us to an entirely new perspective on recommender systems. Namely, how can privacy of the users be protected in recommender systems without loosing too much accuracy?

We describe two solutions that address the problem of preserving privacy of users in recommender systems. In the first approach, user privacy is protected by means of encryption and the recommendations are still generated by processing these encrypted preference values. In the second approach, protecting the privacy of the users is possible without encryption but by means of perturbation of user preference data.

Recommendations by Partial SVD on Encrypted Preferences

Canny [14] addresses the user privacy problem in recommender systems and proposes to encrypt user preferences. Assume that the recommender system applies a collaborative filtering algorithm on a matrix $\mathbf{P}$ of users versus item ratings. Each row of this matrix represents the corresponding user's taste for the corresponding items. Canny proposes to use a collaborative filtering algorithm based on dimension reduction of $\mathbf{P}$. In this way, an approximation matrix of the original preference matrix is obtained in a lower dimension that best represents the user taste for the overall system. When a new user enters the system, the recommendations are generated by simply re-projecting the user preference vector, which has many unrated items, over the approximation matrix. As a result, a new vector will be obtained which contains approximated values for the unrated items [37, 14].

The ratings in recommender systems are usually integer numbers within a small range and items that are not rated are usually assigned to zero. To protect the privacy of the users, the user preferences vector $\mathbf{X} = [x_1, x_2, \ldots, x_M]$ is encrypted individually as $E_{pk}(\mathbf{X})$. To reduce the dimension of the preference matrix $\mathbf{P}$ singular value decomposition (SVD) is an option. The SVD allows $\mathbf{P}$ to be written as:

$$\mathbf{P} = \mathbf{U}\mathbf{D}\mathbf{V}^T, \tag{2.26}$$

where the columns of $\mathbf{U}$ are the left singular vectors, $\mathbf{D}$ is a diagonal matrix containing the singular values, and $\mathbf{V}^T$ has rows that are the right singular vectors.

Once the SVD of the preference matrix $\mathbf{P}$ is computed, an approximation matrix in a lower dimension subspace can be computed easily. Computing the SVD on $\mathbf{P}$ that contains encrypted user preferences is, however, more complicated.

Computing the decomposition of the users' preference matrix requires sums of products of vectors. If the preference vector of each user is encrypted, there is no efficient way of computing sums of products of vectors since this would require an algebraic homomorphic cryptosystem. Using secure multi-party computation protocols on this complex function is costly considering the size of the circuit necessary for the complex operation.

Instead of straightforward computation of SVD, Canny [14] proposed to use an iterative approximation algorithm to obtain a partial decomposition of the user preference matrix. The conjugate gradient algorithm is an iterative procedure consisting merely of *additions* of vectors which *can* be done under homomorphically encrypted user preference vectors. Each iteration in the protocol has two steps: Users compute 1) their contribution to the current gradient and 2) scalar quantities for the optimization of the gradient. Both steps require only additions of vectors thus we only explain the first step.

For the first step of the iterations each user computes his contribution $\mathbf{G}_k$ to the current gradient $\mathbf{G}$ by the following equation:

$$\mathbf{G}_k = \mathbf{A}\mathbf{X}_k^T\mathbf{X}_k(\mathbf{I} - \mathbf{A}^T\mathbf{A}), \tag{2.27}$$

where matrix $\mathbf{A}$ is the approximation of the preference matrix $\mathbf{P}$ and it is initialized as a random matrix before the protocol starts. Each user encrypts his own gradient vector $\mathbf{G}_k$ with the public key of the user group by following the Pedersen's threshold scheme [62] that uses El Gamal cryptosystem which is modified to be additively homomorphic. All contributions from the users are then added up to form the encrypted gradient $E_{pk}(\mathbf{G})$ by using the additive homomorphic property of the cryptosystem,

$$E_{pk}(\mathbf{G}) = E_{pk}\left(\sum_{k \in \text{users}} \mathbf{G}_k\right) = \prod_{k \in \text{users}} E_{pk}(\mathbf{G}_k). \tag{2.28}$$

This resulting vector $E_{pk}(\mathbf{G})$ is then jointly decrypted and used to update the approximated matrix $\mathbf{A}$ which is publicly known and used to compute the new gradient for the next iteration.

Although the protocol is based on addition of vectors, zero-knowledge proof protocols play an important role. The validity of the user inputs, i.e. the encrypted preference vector elements lie in a certain range, are verified by zero-knowledge proofs. Moreover, the partial encryption results from the users are also proved valid by running a zero-knowledge proof protocol. Both group of zero-knowledge proofs are checked by a subgroup of users of whose majority is necessary for the validation.

Canny [15] also applies this approach to a different collaborative filtering method, namely expectation maximization (EM) based factor analysis. Again this algorithm involves simple iterative operations that can be implemented by vector additions. In both recommender system solutions, multiple iterations are necessary for the algorithm to converge and in each iteration users need to participate in the cryptographic computations as in joint decryption and zero-knowledge proofs for input validation. These computations are interactive and thus, it is imperative for the users to be online and synchronized.

Randomized Perturbation to Protect Preferences

Previous section showed that homomorphic cryptosystems, zero-knowledge proof protocols and secure multi-party computations play an important role in providing solutions for processing encrypted data. However, there are other ways to preserve privacy. In the following, we discuss preserving privacy in recommender systems by perturbation of user data.

Randomized perturbation technique was first introduced in privacy preserved data-mining by Agrawal and Srikant [5]. Polat and Du [65, 66] proposed to use this randomization based technique in collaborative filtering. The user privacy is protected by simply randomizing user data while certain computations on aggregate data can still be done. Then, the server generates recommendations based on the blinded data but can not derive the user's private information (Figure 2.5).

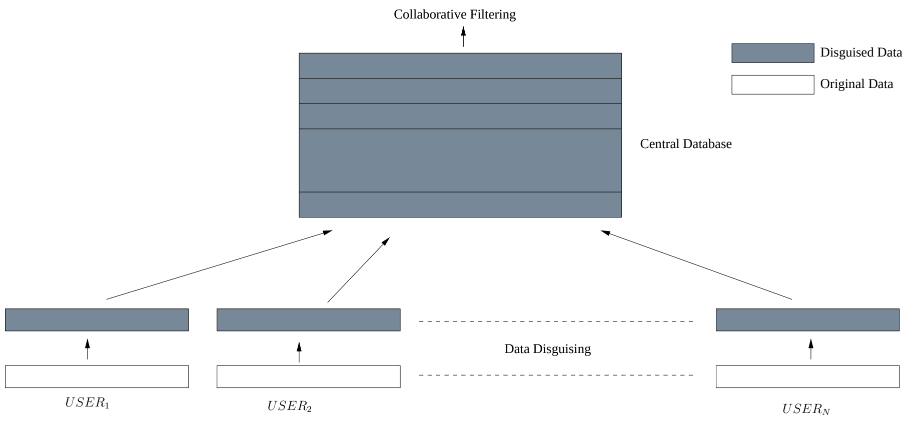


Figure 2.5: Privacy preserving collaborative filtering with user preference perturbation.

Consider the scalar product of two vectors $\mathbf{X}$ and $\mathbf{Y}$. These vectors are blinded by $\mathbf{R} = [r_1, \ldots, r_M]$ and $\mathbf{S} = [s_1, \ldots, s_M]$ such that $\check{\mathbf{X}} = \mathbf{X} + \mathbf{R}$ and $\check{\mathbf{Y}} = \mathbf{Y} + \mathbf{S}$. Here r_i's and s_i's are uniformly distributed random values with zero mean. The scalar product of $\mathbf{X}$ and $\mathbf{Y}$ can be estimated from $\check{\mathbf{X}}$ and $\check{\mathbf{Y}}$:

$$\check{\mathbf{X}} \cdot \check{\mathbf{Y}} = \sum_{k=1}^{M}(x_k y_k + x_k s_k + r_k y_k + r_k s_k) \approx \sum_{k=1}^{M} x_k y_k. \qquad (2.29)$$

Since $\mathbf{R}$ and $\mathbf{S}$ are independent and independent of $\mathbf{X}$ and $\mathbf{Y}$, we have $\sum_{k=1}^{M} x_k s_k \approx 0$, $\sum_{k=1}^{M} r_k y_k \approx 0$, and $\sum_{k=1}^{M} r_k s_k \approx 0$. Similarly, the sum of the elements of any vector $\mathbf{A}$ can be estimated from its randomized form $\mathbf{A}'$. Polat and Du used these two approximations to develop a privacy-preserving collaborative filtering method [65, 66].

This method works if the number of users in the system is significantly large. Only then the computations based on aggregated data can still be computed with sufficient accuracy. Moreover, it is also pointed out in [41, 47] that the idea of preserving privacy by adding random noise might not preserve privacy as much as it had been believed originally. The user data can be reconstructed from the randomly perturbed user data matrix. The main limitation in the original work of Polat and Du is shown to be the item-invariant perturbation [81]. Therefore, Zhang *et al.* [81] propose a two-way communication perturbation scheme for collaborative filtering in which the server and the user communicates to determine perturbation guidance that is used to blind user data before sending to the server. Notwithstanding these approaches, the security of such schemes based on perturbation of data is not well understood.

2.4 Content Protection

2.4.1 Watermarking of Content

In the past decade, content protection measures have been proposed based on digital watermarking technology. Digital watermarking [21, 9] allows hiding into a digital content information that can be detected or extracted at a later moment in time by means of signal processing operations such as correlation. In this way, digital watermarking provides a communication channel multiplexed into original content through which it is possible to transmit information. The type of information transmitted from sender to receiver depends on the application at hand. As an example, in a forensic tracing application, a watermark is used to embed a unique code into each copy of the content to be distributed, where the code links a copy either to a particular user or to a specific device. When unauthorized published content is found, the watermark allows to trace the user who has redistributed the content.

Secure signal processing needs to be performed in case watermark detection or embedding is done in untrusted devices; watermarking schemes usually rely on a symmetric key for both embedding and detection, which is critical to both the robustness and security of the watermark and thus needs to be protected.

For the application of secure signal processing in content protection, three categories can be identified, namely distribution models, customer rights protection, and secure watermark detection. The first two categories are relevant to forensic tracing (fingerprinting) applications. In classical distribution models, the watermark embedding process is carried out by a trusted server before releasing the content to the user. However this approach is not scalable and in large scale distribution systems the server may become overloaded. In addition, since point-to-point communication channels are required, bandwidth requirements become prohibitive. A proposed solution is to use client-side watermark embedding. Since the client is untrusted the watermark needs to be embedded without the client having access to the original content and watermark.

The customer's rights problem relates to the intrinsic problem of ambiguity when watermarks are embedded at the distribution server: a customer whose watermark has been found on unauthorized copies can claim that he has been framed by a malicious seller who inserted his identity as watermark in an arbitrary object. The mere existence of this problem may discredit the accuracy of the forensic tracing architecture. Buyer-seller protocols have been designed to embed a watermark based on the encrypted identity of the buyer, making sure that the watermarked copy is available only to the buyer and not to the seller.

In the watermark detection process, a system has to prove to a verifier that a watermark is present in certain content. Proving the presence of such a watermark is usually done by revealing the required detection information to the verifying party. All current applications assume that the verifier is a trusted party. However, this is not always true, for instance if the prover is a consumer device. A cheating verifier could exploit the knowledge acquired during watermark detection to break the security of the watermarking system. Cryptographic protocols, utilizing zero-knowledge proofs, have been constructed in order to mitigate this problem.

We will first introduce a general digital watermarking model to define the notation that will be useful in the remainder of the section. An example of a watermarking scheme is proposed, namely the one proposed by Cox *et al.* [20], since this scheme is adopted in many of the content protection applications.

Watermarking Model

Figure 2.6 shows a common model for a digital watermarking system [8]. The inputs of the system are the original host signal **X** and some application dependent to-be-hidden information, here represented as a binary string $\mathbf{B} = [b_1, b_2, \ldots, b_L]$, with b_i taking values in $\{0, 1\}$. The embedder inserts the watermark code **B** into the host signal to produce a watermarked signal $\mathbf{X}_w$, usually making use of a secret key sk to control some parameters of the embedding process and allow the recovery of the watermark only to authorized users.

The watermark channel takes into account all processing operations and (intentional or non-intentional) manipulations the watermarked content may undergo during distribution and use. As a result, the watermarked content $\mathbf{X}_w$ is modified into the "received" version $\mathbf{X}'$. Based on $\mathbf{X}'$, either a detector verifies the presence of a specific message given to it as input, thus only answering *yes* or *no*, or a decoder reads the (binary) information conveyed by the watermark. Detectors and decoders may need to know the original content **X** in order to retrieve the hidden information (non-blind detector/decoder), or they do not require the original content (blind or oblivious detector/decoder).

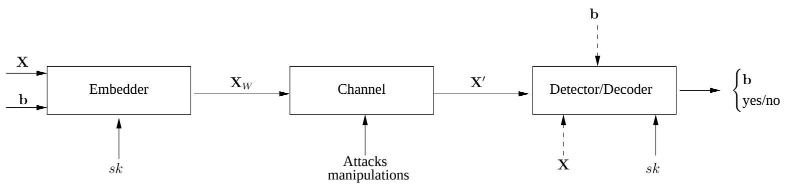


Figure 2.6: A digital watermarking model

Watermarking Algorithm

Watermark information is embedded into host signals by making imperceptual modifications to the host signal. The modifications are such that they convey the to-be-hidden information **B**. The hidden information can be retrieved afterwards from the modified content by detecting the presence of these modifications. Embedding is achieved by modifying the set of features $\mathbf{X} = [x_1, x_2 \dots x_M]$. In the most simple case, the features are simple signal amplitudes. In more complicated scenarios, the features can be DCT or wavelet coefficients. Several watermarking schemes make use of a spread-spectrum approach to code the to-be-hidden information **B** into $\mathbf{W} = [w_1, w_2 \dots w_M]$. Typically, **W** is a realization of a normally distributed random signal with zero mean and unit variance.

The most well-known spread-spectrum techniques was proposed by Cox *et al.* [20]. The host signal is first transformed into a Discrete Cosine Transform (DCT) representation. Next the largest magnitude DCT coefficients are selected, obtaining the set of features **X**. The multiplicative watermark embedding rule is defined as follows:

$$x_{w,i} \quad = \quad x_i + cw_i x_i = x_i(1 + cw_i), \tag{2.30}$$

where $x_{w,i}$ is the i-th component of the watermarked feature vector and c is a scaling factor controlling the watermark strength. Finally, an inverse DCT transform yields the watermarked signal $\mathbf{X}_w$.

To determine if a given signal **Y** contains the watermark **W**, the decoder computes the DCT of **Y**, extracts the set $\mathbf{X}'$ of largest DCT coefficients, and then computes the correlation $\rho_{\mathbf{X}'\mathbf{W}}$ between the features $\mathbf{X}'$ and the watermark **W**. If the correlation is larger than a threshold δ, i.e.,

$$\rho_{\mathbf{X}'\mathbf{W}} \quad = \quad \frac{<\mathbf{X}', \mathbf{W}>}{<\mathbf{X}', \mathbf{X}'>} \geq \delta, \tag{2.31}$$

the watermark is considered present in **Y**.

2.4.2 Client-side Watermark Embedding

Client-side watermark embedding systems transmit the same encrypted version of the original content to all the clients but a client-specific decryption key allows to decrypt the content and at the same time implicitly embed a watermark. When the client uses his key to decrypt the content, he obtains a uniquely watermarked version of the content. The security properties of the embedding scheme usually guarantees that obtaining either the watermark or the original content in the clear is of comparable hardness as removing the watermark from the personalized copy.

In literature, several approaches for secure embedding can be found. In [31] a pseudorandom mask is blended over each frame of a video. Each client is given a different mask, which, when subtracted from the masked broadcast video, leaves an additive watermark in the content. The scheme is not very secure because since the same mask is used for all frames of a video, it can be estimated by averaging attacks.

In broadcast environments, stream switching [24, 59] can be performed. Two differently watermarked signals are chopped up into small chunks. Each chunk is encrypted by a different key. Clients are given a different set of decryption keys that allow them to selectively decrypt chunks of the two broadcast streams such that each client obtains the full stream decrypted. The way the full stream is composed out of the two broadcast versions encodes the watermark. This solution consumes considerable bandwidth, since the data to be broadcast to the clients is twice as large as the content itself.

A second solution involves partial encryption, for instance encrypting the signs of DCT coefficients of a signal [48]. Since the sign bits of DCT coefficients are perceptually significant, the partially encrypted version of the signal is heavily distorted. During decryption each user has a different keys that decrypts only a subset of these coefficients, so that some signs are left unchanged. This leaves a detectable fingerprint in the signal. A similar approach was used in [51] to obtain partial encryption-based secure embedding solutions for audio-visual content.

A third approach is represented by methods using a stream-cipher that allows the use of multiple decryption keys, which decrypt the same cipher-text to slightly different plain-texts. Again, the difference between the original and the decrypted content represents the embedded watermark. The first scheme following this approach was proposed by Anderson *et al.* [7] who designed a special stream cipher, called Chameleon, which allows to decrypt Chameleon-encrypted content in slightly different ways. During encryption, a key and a secure index generator are used to generate a sequence of indices, which are used to select four entries from a look-up-table (LUT). These entries are XORed with the plaintext to form a word of the ciphertext. The decryption process is identical to encryption except for the use of a decryption LUT, which is obtained by properly inserting bit errors in some entries of the encryption LUT. Decryption superimposes these errors onto the content, thus leaving a unique watermark. Recently, Adelsbach *et al.* [1] and Celik *et al.* [16] proposed generalizations of Chameleon, suitable for embedding robust spread spectrum watermarks. The schemes operate on lookup-tables composed of integers from $\mathbb{Z}_p$ and replace the XOR operation by a (modular) addition.

In more detail, the secure embedding solution works as follows. The distribution server generates a long-term master encryption look-up table (LUT) **E** of size L, whose entries are properly generated random samples; **E** will be used to encrypt the content to be distributed to the clients. Next, for the k-th client, the server generates a personalized watermark LUT $\mathbf{W}_k$ according to a desired probability distribution, and builds a personalized decryption LUT $\mathbf{D}_k$ by combining the master LUT and the watermark LUT:

$$\mathbf{D}_k[i] = -\mathbf{E}[i] + \mathbf{W}_k[i]. \tag{2.32}$$

The personalized LUTs are then transmitted once to each client over a secure channel. Let us note that the generation of the LUTs is carried out just once at the setup of the application. A content $\mathbf{X}$ is encrypted by adding to it a pseudo-random sequence obtained by selecting some entries of the LUT with a secure pseudo-random sequence generator driven by a session key sk. Each client receives the encrypted content $\mathbf{X}'$ along with the session key sk and decrypts it using some entries of his/her personalized decryption LUT $\mathbf{D}_k$ (again chosen according to sk), with the final effect that a spread-spectrum watermark sequence is embedded into the decrypted content. This process is summarized in Figure 2.7. In detail, driven by the session key sk, a set of

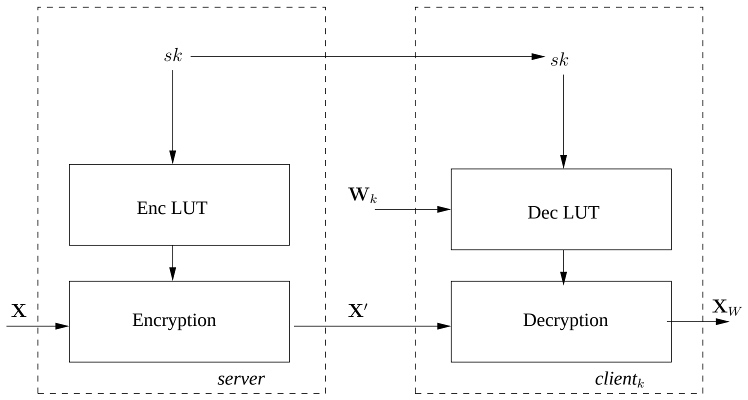


Figure 2.7: Encryption and following joint decryption and watermarking procedure proposed in [16].

indices t_{ij} is generated, where $0 \leq i \leq M-1$, $0 \leq j \leq S-1$, $0 \leq t_{ij} \leq L-1$. Each feature of the content x_i is encrypted by adding S entries of the encryption LUT, obtaining the encrypted feature x'_i as follows:

$$x'_i = x_i + \sum_{j=0}^{S-1} \mathbf{E}[t_{ij}]. \tag{2.33}$$

Joint decryption and watermarking is accomplished by reconstructing with the session key sk the same set of indices t_{ij} and by adding S entries of the decryption LUT to each encrypted feature x'_i:

$$x_{w,i} = x'_i + \sum_{j=0}^{S-1} \mathbf{D}[t_{ij}] = x_i + \sum_{j=0}^{S-1} \mathbf{W}[t_{ij}] = x_i + w_i. \tag{2.34}$$

2.4.3 Buyer Seller Protocols

Forensic tracing architectures which perform watermark embedding at the distribution server are vulnerable against a dishonest seller. The mere fact that a seller may fool a buyer may have an impact on the credibility of the whole tracing system. (Note that a seller may in fact have an incentive to fool a buyer: a seller who acts as an authorized re-selling agent may be interested in distributing many copies of a work containing the fingerprint of a single buyer to avoid paying the royalties to the author, by claiming that such copies were illegally distributed or sold by the buyer).

A possible solution consists in resorting to a trusted third party, responsible for both embedding and detection of watermarks; however, such an approach is not feasible in practical applications, because the TTP could easily become a bottleneck for the whole system. The Buyer-Seller Protocol relies on cryptographic primitives to perform watermark embedding [55]; the protocol assures that the seller does not have access to the watermarked copy carrying the identity of the buyer, hence he cannot distribute or sell these copies. In spite of this, the seller can identify the buyer from whom unauthorized copies originated, and prove it by using a proper dispute resolution protocol.

We describe the protocol by Memon and Wong [55] in more detail. Let Alice be the seller, Bob the buyer, and WCA a trusted watermark certification authority in charge of generating legal watermarks and sending them to any buyer upon request. The protocol uses a public key cryptosystem which is homomorphic with respect to the operation used in the watermark embedding equation (i.e., the cryptosystem will be multiplicatively homomorphic if watermark embedding is multiplicative, like in Cox's scheme); moreover, Alice and Bob possess a pair of public/private keys denoted by pk_A, pk_B (public keys) and sk_A, sk_B (private keys).

In the first part of the protocol, on request of Bob, the WCA generates a valid watermark signal $\mathbf{W}$ and sends it back to Bob, encrypted with Bob's public key $E_{pk_B}(\mathbf{W})$, along with its digital signature $S_{WCA}(E_{pk_B}(\mathbf{W}))$, to prove that the watermark is valid.

Next, Bob sends to Alice $E_{pk_B}(\mathbf{W})$ and $S_{WCA}(E_{pk_B}(\mathbf{W}))$, so that Alice can verify that the encrypted watermark has been generated by the WCA. Alice performs two watermark embedding operations. First, she embeds (with any watermarking scheme) into the original content $\mathbf{X}$ a watermark $\mathbf{V}$, which just conveys a distinct ID univocally identifying the transaction, obtaining the watermarked content $\mathbf{X}_w$. Next, a second watermark is built by using $E_{pk_B}(\mathbf{W})$: Alice permutes the watermark components through a secret permutation π

$$\pi(E_{pk_B}(\mathbf{W})) \quad = \quad E_{pk_B}(\pi(\mathbf{W})), \tag{2.35}$$

and inserts $E_{pk_B}(\pi(\mathbf{W}))$ in $\mathbf{X}_w$ directly in the encrypted domain, obtaining the final watermarked content $\mathbf{X}''$ in encrypted form; $\mathbf{X}''$ is thus unknown to her. This is possible due to the homomorphic property of the cipher:

$$E_{pk_B}(\mathbf{X}'') \quad = \quad E_{pk_B}(\mathbf{X}_w) \cdot E_{pk_B}(\pi(\mathbf{W})). \tag{2.36}$$

When Bob receives $E_{pk_B}(\mathbf{X}'')$, he decrypts it by using his private key sk_B, thus obtaining $\mathbf{X}''$, where the watermarks $\mathbf{V}$ and $\pi(\mathbf{W})$ are embedded. Note that Bob cannot read the watermark $\pi(\mathbf{W})$, since he does not know the permutation π. The scheme is represented in Figure 2.8.

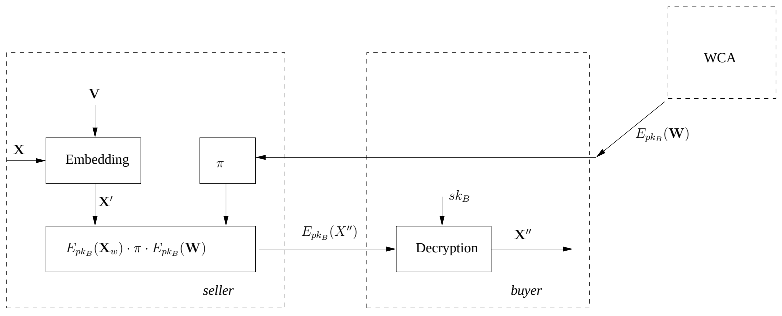


Figure 2.8: The scheme of the Buyer Seller protocol proposed in [55].

In order to recover the identity of potential copyright violators, Alice first looks for the presence of $\mathbf{V}$. Upon detection of an unauthorized copy of $\mathbf{X}$, say $\mathbf{Y}$, she can use the second watermark to effectively prove that the copy originated from Bob. To do so, Alice must reveal to a judge the permutation π, the encrypted watermark $E_{pk_B}(\mathbf{W})$ and $S_{WCA}(E_{pk_B}(\mathbf{W}))$. After verifying $S_{WCA}(E_{pk_B}(\mathbf{W}))$, the judge asks Bob to use his private key sk_B to compute and reveal $\mathbf{W}$. Now it is possible to check $\mathbf{Y}$ for the presence of $\pi(\mathbf{W})$: if such a presence is verified, then Bob is judged guilty, otherwise Bob's innocence has been proven. Note that if $\pi(\mathbf{W})$ is found in $\mathbf{Y}$, Bob can not state that $\mathbf{Y}$ originated from Alice, since to do so Alice should have known either $\mathbf{W}$ to insert it within the plain asset $\mathbf{X}$, or sk_B to decrypt $E_{pk_B}(\mathbf{X}'')$ after the watermark was embedded in the encrypted domain.

As a particular implementation of the protocol, [55] proposed to use Cox's watermarking scheme and a multiplicatively homomorphic cipher (despite its deterministic nature, authors use RSA). More secure and less complex implementations of the Buyer Seller protocol have been proposed in [50, 80, 49, 6].

2.4.4 Secure Watermark Detection

To tackle the problem of watermark detection in the presence of an untrusted verifier (to whom the watermark secrets cannot be disclosed), two approaches have been proposed: one approach called *asymmetric watermarking* [34, 30] uses different keys for watermark embedding and detection. Whereas a watermark is embedded using a private key, its presence can be detected by a public key. In such schemes, the knowledge of the public detection key must not enable an adversary to remove the embedded watermark; unfortunately, none of the proposed schemes is sufficiently robust against malicious attacks [29]. Another approach is represented by *zero-knowledge watermark detection*.

Zero-knowledge watermark detection (ZKWD) uses a cryptographic protocol to wrap a standard symmetric watermark detection process. In general, a zero-knowledge watermark detection algorithm is an interactive proof system where a prover tries to convince a verifier that a digital content $\mathbf{X}'$ is watermarked with a given watermark **B** without disclosing **B**. In contrast to the standard watermark detector, in ZKWD the verifier is given only properly encoded (or encrypted) versions of security-critical watermark parameters. Depending on the particular protocol, the watermark code, the watermarked object, a watermark key or even the original unmarked object is available in an encrypted form to the verifier. The prover runs the zero-knowledge watermark detector to demonstrate to the verifier that the encoded watermark is present in the object in question, without removing the encoding. A protocol run will not leak any information except for the unencoded inputs and the watermark presence detection result.

Early approaches for zero-knowledge watermark detection used permutations to conceal both the watermark and the object [23]; the protocol assures that the permuted watermark is detected in the permuted content and that both the watermark and the object are permuted in the same manner. Craver [22] proposed to use ambiguity attacks as a central tool to construct zero-knowledge detectors; such attacks allow to compute a watermark that is detectable in a content but never has been embedded there. To use ambiguity attacks in a secure detector, the real watermark is concealed within a number of fake marks. The prover has to show that there is a valid watermark in this list without revealing its position. Now, the adversary (equipped solely with a watermark detector) cannot decide which of the watermarks is not counterfeit. Removal of the watermark is thus sufficiently more difficult.

Another proposal is to compute the watermark detection statistic in the encrypted domain (e.g., by using additive homomorphic public-key encryption schemes or commitments) and then use zero-knowledge proofs to convince the verifier that the detection statistic exceeds a fixed threshold. This approach was first proposed by Adelsbach and Sadeghi [3], who use a homomorphic commitment scheme to compute the detection statistic; the approach was later refined in [2].

Adelsbach and Sadeghi [3] propose a zero-knowledge protocol based on the Cox's watermarking scheme. In contrast to the original algorithm, it is assumed that the watermark and DCT-coefficients are integers and not real numbers (this can be achieved by appropriate quantization). Moreover, for efficiency reasons the correlation computation in Eq. (2.31) is replaced by the detection criterion:

$$\begin{aligned} C &:= (<\mathbf{X}',\mathbf{W}>)^2 - <\mathbf{X}',\mathbf{X}'> \cdot\, \delta^2 \\ &:= (A)^2 - B \geq 0; \end{aligned} \tag{2.37}$$

the latter detection criterion is equivalent to the original one, provided that the factor A is positive.

The following Zero-Knowledge Detection Protocol has been designed to allow the prover to prove to a verifier that the watermark committed to in the commitment $com(\mathbf{W})$ is present in the received content $\mathbf{X}'$, without revealing any information about **W**. In the protocol, the authors employ an additively homomorphic commitment scheme (such as the one proposed by Damgård and Fujisaki [25]). Let p_{pub}, $\mathbf{X}'$, $com(\mathbf{W})$, δ be the common inputs of prover and verifier and let p_{sec} be the private input of the prover. First, both prover and verifier select the watermarked features $\mathbf{X}'$ and compute the value B of Eq. (2.37); the prover sends a commitment $com(B)$ to the verifier and opens it immediately, allowing him to verify that the opened commitment contains the same value B he computed himself. Now both compute the commitment

$$com(A) = \prod_{i=1}^{M} com(w_i)^{x'_i}, \tag{2.38}$$

by taking advantage of the homomorphic property of the commitment scheme. Subsequently the prover proves in zero-knowledge that $A \geq 0$. Next, the prover computes the value A^2, sends a commitment $com(A^2)$ to the verifier and gives him a zero-knowledge proof that it really contains the square of the value contained in $com(A)$. Being convinced that $com(A^2)$ really contains the correctly computed value A^2, the two parties compute the commitment $com(C) := com(A^2)/com(B)$ on the value C. Finally the prover proves to the verifier, with a proper zero-knowledge protocol, that $com(C) \geq 0$. If this proof is accepted then the detection algorithm ends with true, otherwise with false.

While early protocols addressed only correlation-based watermark detectors, the approach has recently be extended to Generalized Gaussian Maximum Likelihood detectors [75] and Dither Modulation watermarks [64, 54].

2.5 Problem Statement

This chapter's comprehensive study on privacy enhanced solutions for different problem domains has shown important insights about preserving privacy in multimedia applications. First, the applications that we encounter are from the field of signal processing and thus, they often consist of similar operations. Second, the data in these applications have signal properties, meaning that they are values in a small range. Third, processing encrypted data by using techniques and tools from cryptography seems to be feasible if the signal aspects of the data are considered. However, currently available cryptographic tools and protocols are mostly generic and not designed by considering the features of the data. If applied directly for multimedia applications, the resulting systems will be inefficient. This observation leads us into a new direction in which we exploit the application requirements and the structure of the data to develop privacy enhanced solutions.

The goal of this thesis is to present a methodology for preserving privacy in multimedia applications. For this purpose, we focus on cryptographic tools, in particular homomorphism and MPC techniques. The designed systems must satisfy three major requirements, namely:

- **Correctness.** The privacy enhanced system must have identical or similar outputs compared to the original system that works on plain text signals such that the users of the service cannot see a difference.

- **Privacy.** No party should gain information on the input of the other parties. This requirement involves the intermediate values of the algorithms which can leak information on the input of parties.

- **Efficiency.** The overhead introduced by using cryptographic tools to provide privacy must be minimum in terms of communication and computation costs and, if applicable, storage capacity.

Correctness and privacy can be verified once the protocols are composed, however achieving efficiency presents the main challenge in this thesis. In order to obtain minimum overhead, we propose to exploit the signal properties of the data in multimedia applications. For this purpose, we have selected a number of prototypical applications that possess commonalities with respect to signal processing operations and propose privacy-preserving solutions for each application based on homomorphism and MPC techniques over integer arithmetic in a semi-honest model. The following challenges, which are vital to obtain a complete privacy-preserving version of the applications, are addressed and efficient solutions are proposed.

Data Representation

The data we consider in this thesis are signal samples such as images, preferences and feature vectors which often consist of small and integer values. However, throughout the signal processing, the size and the type of the data can change. For instance, the DCT of an image block usually consists of real values whose ranges are larger than the 8-bit pixel value. Similarly, the distance and correlation computations change the bit length and the type of the data, respectively. As we propose to protect the privacy-sensitive data by means of encryption and we use semantically secure homomorphic cryptosystems which operate only on integers, it is mandatory to come up with a strategy that copes with the increase in data bit size and possible changes in data type.

Linear and Non-linear Operations

The homomorphic encryption allows us to implement certain linear operations on encrypted data. However, minimizing the number of operations on the encrypted data plays a crucial role in designing privacy enhanced multimedia applications with less computational power requirement.

While homomorphic encryption schemes allow us to realize linear operations in the encrypted domain, cryptographic protocols based on MPC techniques are necessary for the non-linear operations. Currently available protocols are mostly generic and they do not consider the structure of the data or the application. Thus, cryptographic protocols for realizing non-linear operations for signal processing applications are needed.

Data Expansion

With homomorphic cryptosystems, we encrypt individual signal samples and process them in the encrypted domain later on. As discussed in this chapter, we also need semantic security as the bit size of signal samples considered in signal processing applications are rather small. Encrypting small signal samples by using semantically secure encryption schemes results in data expansion. This expansion for an 8-bit signal sample is by a factor of 256 after the encryption with a modest key size of

1024 bits in the Paillier cryptosystem. As we have data in the order of megabytes and gigabytes, this data expansion will introduce a substantial load in communication and storage. Thus, an effective approach to minimize the effect of data expansion is required.

Computation and Communication Costs

A final challenge of working in the encrypted domain is minimizing the computation power and bandwidth requirements. As we operate on encrypted data, which are in the order of thousand bits, each operation will consume time. Table 2.5 shows the average run time for several operations under Paillier encryption scheme with a key size of 1024 bits (the cipher text space is 2048 bits). The message and the public value are 100 bits each. As the key length is increased for security reasons, the time consumption of operations also increases (Figure 2.9). Considering that these operations are repeated for many times, the overall time consumption of the proposed protocols will be a major problem. For example, encrypting an 8-bit gray scale image of size 840×600 pixels will take roughly 55 minutes.

In addition to computational costs, increase in the communication cost due to data expansion after encryption presents a major challenge. To illustrate the affect of encryption on data size, consider that an 8-bit gray scale image of size 840×600 pixels becomes 123 MB after encrypting each pixel value in Paillier encryption scheme with a key size of 1024 bits. Thus, strategies to minimize the number of operations and the cost of communication are imperative.

Table 2.2: Average time consumption for various operations for Paillier cryptosystem on a Pentium Xeon, 2.33 GHz machine.

Operation	**Time**
Encryption: $(c = E_{pk}(m))$	6690 μs
Decryption: $(m = D_{sk}(c))$	6650 μs
Multiplying two encrypted values: $(E_{pk}(m_1) \cdot E_{pk}(m_2))$	7.1 μs
Raising an encrypted value to the power of a public value: $(E_{pk}(m)^c)$	710 μs

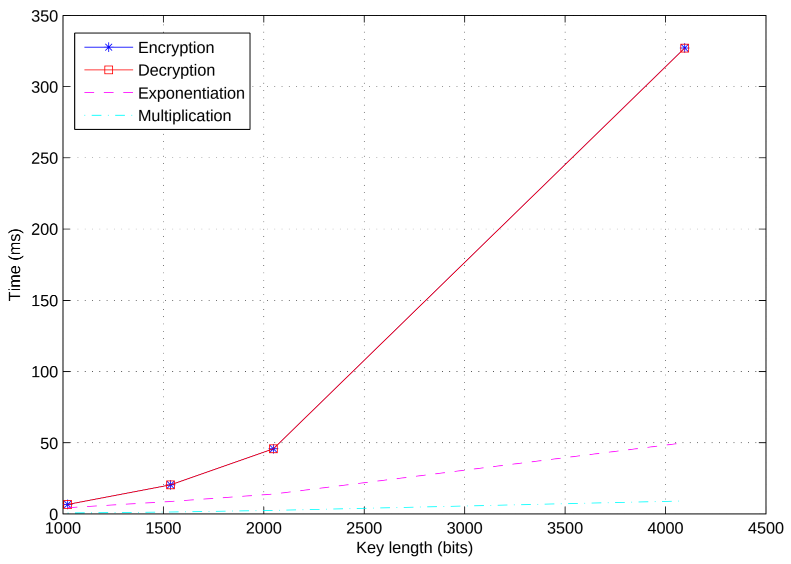


Figure 2.9: Run time of operations for the Paillier scheme with different key lengths.

References

[1] A. Adelsbach, U. Huber, and A.-R. Sadeghi. Fingercasting—joint fingerprinting and decryption of broadcast messages. In *11th Australasian Conference on Information Security and Privacy*, volume 4058 of *Lecture Notes in Computer Science*, pages 136–147. Springer, 2006.

[2] A. Adelsbach, M. Rohe, and A.-R. Sadeghi. Non-interactive watermark detection for a correlation-based watermarking scheme. In *Communications and Multimedia Security*, volume 3677 of *Lecture Notes in Computer Science*, pages 129–139. Springer, 2005.

[3] A. Adelsbach and A.-R. Sadeghi. Zero-knowledge watermark detection and proof of ownership. In *4th International Workshop on Information Hiding—IHW 2001*, volume 2137 of *Lecture Notes in Computer Science*, pages 273–288, Pittsburgh, PA, USA, 2001. Springer Verlag.

[4] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order preserving encryption for numeric data. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data —SIGMOD'04*, pages 563–574, New York, NY, USA, 2004. ACM Press.

[5] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data — SIGMOD'00:*, volume 29(2), pages 439–450. ACM Press New York, NY, USA, 2000.

[6] F. Ahmed, F. Sattar, M. Y. Siyal, and D. Yu. A secure watermarking scheme for buyer-seller identification and copyright protection. *EURASIP Journal on Applied Signal Processing*, page 15, 2006.

[7] R. J. Anderson and C. Manifavas. Chameleon—a new kind of stream cipher. In *Proceedings of the 4th International Workshop on Fast Software Encryption — FSE'97*, pages 107–113, London, UK, 1997. Springer-Verlag.

[8] M. Barni and F. Bartolini. Data hiding for fighting piracy. *IEEE Signal Processing Magazine*, 21(2):28 – 39, 2004.

[9] M. Barni and F. Bartolini. *Watermarking Systems Engineering: Enabling Digital Assets Security and Other Applications*. Marcel Dekker, 2004.

[10] J. Benaloh. *Verifiable secret-ballot elections*. PhD thesis, Yale University, Department of Computer Science, 1988.

[11] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. *Advances in Cryptology–EUROCRYPT*, 3027:506–522, 2004.

[12] F. Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 431–444. Springer Verlag, 2000.

[13] J. Camenisch and M. Michels. Proving in zero-knowledge that a number is the product of two safe primes. In *Advances in Cryptology—EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 107–122. Springer Verlag, 1999.

[14] J. F. Canny. Collaborative filtering with privacy. In *IEEE Symposium on Security and Privacy*, pages 45–57, 2002.

[15] J. F. Canny. Collaborative filtering with privacy via factor analysis. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 238–245, New York, NY, USA, 2002. ACM Press.

[16] M. Celik, A. Lemma, S. Katzenbeisser, and M. van der Veen. Secure embedding of spread-spectrum watermarks using look-up tables. In *International Conference on Acoustics, Speech and Signal Processing —ICASSP'07*. IEEE Press, 2007.

[17] J. H. Cheon and H. S. Nam. A cryptanalysis of the original Domingo-Ferrer's algebraic privacy homomophism. Cryptology ePrint Archive, Report 2003/221, 2003.

[18] S.-C. S. Cheung and T. Nguyen. Secure multiparty computation between distrusted networks terminals. *EURASIP Journal on Information Security*, page 10, 2007.

[19] C. Clifton, M. Kantarcioglu, X. Lin, J. Vaidya, and M. Zhu. Tools for privacy preserving distributed data mining. *ACM SIGKDD Explorations Newsletter*, 4(2):28–34, Dec. 2002.

[20] I. Cox, J. Kilian, T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12):1673–1687, Dec. 1997.

[21] I. J. Cox, M. L. Miller, and J. A. Bloom. *Digital Watermarking*. Morgan Kaufmann, 2001.

[22] S. Craver. Zero knowledge watermark detection. In *Information Hiding — 3rd International Workshop, IH'99*, volume 1768 of *Lecture Notes in Computer Science*, pages 101–116. Springer Verlag, 1999.

[23] S. Craver and S. Katzenbeisser. Security analysis of public-key watermarking schemes. In *Proceedings of SPIE, Mathematics of Data/Image Coding, Compression and Encryption IV, with Applications*, volume 4475, pages 172–182, 2001.

[24] J. Crowcroft, C. Perkins, and I. Brown. A method and apparatus for generating multiple watermarked copies of an information signal. WO Patent No. 00/56059, 2000.

[25] I. Damgård and E. Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In Y. Zheng, editor, *Advances in Cryptology—ASIACRYPT'02*, volume 2501 of *Lecture Notes in Computer Science*, pages 125–142. Springer, 2002.

[26] I. Damgård and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In *Public Key Cryptography*, pages 119–136, 2001.

[27] J. Domingo-Ferrer. A new privacy homomorphism and applications. *Information Processing Letters*, 60(5):277–282, 1996.

[28] J. Domingo-Ferrer. A provably secure additive and multiplicative privacy homomorphism. In *Proceedings of the 5th International Conference on Information Security —ISC '02*, number 2433 in Lecture Notes in Computer Science, pages 471–483. Springer-Verlag, 2002.

[29] J. J. Eggers, J. K. Su, and B. Girod. Asymmetric watermarking schemes. In *Sicherheit in Mediendaten, GMD Jahrestagung, Proceedings*, 2000.

[30] J. J. Eggers, J. K. Su, and B. Girod. Public key watermarking by eigenvectors of linear transforms. In *Proceedings of the European Signal Processing Conference*, 2000.

[31] S. Emmanuel and M. Kankanhalli. Copyright protection for MPEG-2 compressed broadcast video. In *IEEE International Conference on Multimedia and Expo. — ICME 2001*, pages 206–209, 2001.

[32] C. Fontaine and F. Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security*, 2007:10, 2007.

[33] E. Fujisaki and T. Okamoto. Statistical zero-knowledge protocols to prove modular polynomial relations. In *Advances in Cryptology — CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30. Springer, 1997.

[34] T. Furon and P. Duhamel. An asymmetric public detection watermarking technique. In *Information Hiding, 3rd International Workshop*, number 1768 in Lecture Notes in Computer Science, pages 88–100. Springer, 2000.

[35] T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology, Proceedings of CRYPTO'84*, number 196 in Springer Lecture Notes in Computer Science, pages 10–18, 1984.

[36] B. Goethals, S. Laur, H. Lipmaa, , and T. Mielikainen. On secure scalar product computation for privacy-preserving data mining. In *7th International Conference on Information Security and Cryptology —ICISC'04*, 2004.

[37] K. Y. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[38] O. Goldreich. *Foundations of Cryptography I*. Cambridge University Press, 2001.

[39] O. Goldreich. *Foundations of Cryptography II*. Cambridge University Press, 2004.

[40] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[41] Z. Huang, W. Du, and B. Chen. Deriving private information from randomized data. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data—SIGMOD'05*, pages 37–48. ACM Press New York, NY, USA, 2005.

[42] G. Jagannathan and R. N. Wright. Privacy-preserving distributed K-means clustering over arbitrarily partitioned data. In *KDD'05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 593–599, New York, NY, USA, 2005. ACM Press.

[43] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[44] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran. On compressing encrypted data. *IEEE Trans. on Signal Processing*, 52(10):2992–3006, October 2004.

[45] JPSEC. International standard, ISO/IEC 15444-8:2007.

[46] M. Kantarcioglu and J. Vaidya. Privacy preserving naive Bayes classifier for horizontally partitioned data. In *IEEE ICDM Workshop on Privacy Preserving Data Mining*, pages 3–9, 2003.

[47] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. *Third IEEE International Conference on Data Mining —ICDM'03*, pages 99–106, 2003.

[48] D. Kundur. Video fingerprinting and encryption principles for digital rights management. *Proceedings of the IEEE*, 92(6):918–932, 2004.

[49] M. Kuribayashi and H. Tanaka. Fingerprinting protocol for images based on additive homomorphic property. *IEEE Transactions on Image Processing*, 14(12):2129–2139, December 2005.

[50] C.-L. Lei, P.-L. Yu, P.-L. Tsai, and M.-H. Chan. An efficient and anonymous buyer-seller watermarking protocol. *IEEE Transactions on Image Processing*, 13(12):1618–1626, December 2004.

[51] A. Lemma, S. Katzenbeisser, M. Celik, and M. van der Veen. Secure watermark embedding through partial encryption. In *International Workshop on Digital Watermarking (IWDW)*, volume 4283 of *Springer Lecture Notes in Computer Science*, pages 433–445, 2006.

[52] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology—CRYPTO'00*, volume 1880 of *Lecture Notes in Computer Science*, pages 36–54, 2000.

[53] H. Lipmaa. On diophantine complexity and statistical zero-knowledge arguments. In C. S. Laih, editor, *Advances on Cryptology—ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 398–415. Springer-Verlag, 2003.

[54] M. Malkin and T. Kalker. A cryptographic method for secure watermark detection. In *Proc. 8th Int. Work. on Information Hiding—IH'06*, Lecture Notes in Computer Science, pages 26–41, Old Town Alexandria, Virginia, USA, 10-12 July 2006. Springer Verlag.

[55] N. Memon and P. Wong. A buyer-seller watermarking protocol. *IEEE Transactions on Image Processing*, 10(4):643–649, April 2001.

[56] D. Naccache and J. Stern. A new public-key cryptosystem based on higher residues. In *ACM Conference on Computer and Communications Security*, pages 59–66, 1998.

[57] T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Advances in Cryptology — EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 308–318. Springer, 1998.

[58] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology—EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.

[59] R. Parviainen and P. Parnes. Large scale distributed watermarking of multicast media through encryption. In *Proceedings of the International Federation for Information Processing, Communications and Multimedia Security Joint working conference IFIP TC6 and TC11*, pages 149–158, 2001.

[60] J.-R. T. Pastoriza, S. Katzenbeisser, M. Celik, and A. Lemma. A secure multidimensional point inclusion protocol. In *ACM Multimedia and Security Workshop—MMSEC'07*, pages 109–120. ACM Press New York, NY, USA, 2007.

[61] T. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology — CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.

[62] T. Pedersen. A threshold cryptosystem without a trusted party (extended abstract). In *Advances in Cryptology — EUROCRYPT'91*, volume 547 of *Lecture Notes in Computer Science*, pages 522–526, 1991.

[63] B. Pinkas. Cryptographic techniques for privacy-preserving data mining. *ACM Special Interest Group on Knowledge Discovery and Data Mining SIGKDD Explorations Newsletter*, 4(2):12–19, 2002.

[64] A. Piva, V. Cappellini, D. Corazzi, A. D. Rosa, C. Orlandi, and M. Barni. Zero-knowledge ST-DM watermarking. In *Security, Steganography, and Watermarking of Multimedia Contents VIII*, pages 291–301. SPIE, 2006.

[65] H. Polat and W. Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *Third IEEE International Conference on Data Mining—ICDM'03*, pages 625–628. IEEE Computer Society, 2003.

[66] H. Polat and W. Du. SVD-based collaborative filtering with privacy. In *Proceedings of the 2005 ACM symposium on Applied computing—SAC '05*, pages 791–795, New York, NY, USA, 2005. ACM Press.

[67] S. Pradhan and K. Ramchandran. Distributed source coding using syndromes (DISCUS): design and construction. *IEEE Transactions on Information Theory*, 49(3):626–643, 2003.

[68] J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd International Conference on Machine Learning—ICML'05*, pages 713–719, New York, NY, USA, 2005. ACM Press.

[69] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[70] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender systems. *ACM WebKDD Workshop*, 2000.

[71] C. P. Schnorr. Efficient identification and signatures for smart cards. In *Advances in Cryptology — CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer, 1990.

[72] B. Schoenmakers and P. Tuyls. Practical two-party computation based on the conditional gate. In *Advances in Cryptology—ASIACRYPT'04*, number 3329 in Lecture Notes in Computer Science, pages 119–136. Springer, 2004.

[73] B. Schoenmakers and P. Tuyls. Efficient binary conversion for Paillier encrypted values. In *Advances in Cryptology —EUROCRYPT'06*, volume 4004 of *Lecture Notes in Computer Science*, pages 522–537. Springer, 2006.

[74] D. Slepian and J. K. Wolf. Noiseless coding of correlated information sources. *IEEE Transactions on Information Theory*, 19:471–480, 1973.

[75] J. R. Troncoso and F. Perez-Gonzalez. Efficient non-interactive zero-knowledge watermark detector robust to sensitivity attacks. In P. W. Wong and E. J. Delp, editors, *Security, Steganography, and Watermarking of Multimedia Contents IX, Proc. SPIE Vol. 6505*, page 12, San Jose, CA, USA, January 2007.

[76] V. Verykios, E. Bertino, I. Fovino, L. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-art in privacy preserving data mining. *ACM SIGMOD Record*, 33(1):50–57, 2004.

[77] D. Wagner. Cryptanalysis of an algebraic privacy homomorphism. In *Proceedings of the 6th International Conference on Information Security—ISC 2003*, number 1109 in Lecture Notes in Computer Science, pages 234–239. Springer-Verlag, 2003.

[78] A. Yao. How to generate and exchange secrets. In *IEEE Symposium on Foundations of Computer Science—FOCS'86*, pages 162–167, 1986.

[79] A. C. Yao. Protocols for secure computations. In *Proceedings of Twenty-third IEEE Symposium on Foundations of Computer Science*, pages 160–164, Chicago, Illinois, November 1982.

[80] J. Zhang, W. Kou, and K. Fan. Secure buyer-seller watermarking protocol. *IEE Proceedings on Information Security*, 153(1):15–18, March 2006.

[81] S. Zhang, J. Ford, and F. Makedon. A privacy-preserving collaborative filtering scheme with two-way communication. In *Proceedings of the 7th ACM conference on Electronic commerce—EC'06*, pages 316–323, New York, NY, USA, 2006. ACM Press.

Three

Privacy-Preserving Face Recognition

This chapter has been published as "Privacy-Preserving Face Recognition", by Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, R. L. Lagendijk and T. Toft in the Proceedings of 9^{th} *Symposium on Privacy Enhanced Techonlogies*, pages 235-253, 2009.

Abstract

Face recognition is increasingly deployed as a means to unobtrusively verify the identity of people. The widespread use of biometrics raises important privacy concerns, in particular if the biometric matching process is performed at a central or untrusted server, and calls for the implementation of Privacy-Enhancing Technologies. In this paper we propose for the first time a strongly privacy-enhanced face recognition system, which allows to efficiently hide both the biometrics and the result from the server that performs the matching operation, by using techniques from secure multiparty computation. We consider a scenario where one party provides a face image, while another party has access to a database of facial templates. Our protocol allows to jointly run the standard Eigenfaces recognition algorithm in such a way that the first party cannot learn from the execution of the protocol more than basic parameters of the database, while the second party does not learn the input image or the result of the recognition process. At the core of our protocol lies an efficient protocol for securely comparing two Pailler-encrypted numbers. We show through extensive experiments that the system can be run efficiently on conventional hardware.

3.1 Introduction

Biometric techniques have advanced over the past years to a reliable means of authentication, which are increasingly deployed in various application domains. In particular, face recognition has been a focus of the research community due to its unobtrusiveness and ease of use: no special sensors are necessary and readily available images of good quality can be used for biometric authentication. The development of new biometric face-recognition systems was mainly driven by two application scenarios:

- To reduce the risk of counterfeiting, modern electronic passports and identification cards contain a chip that stores information about the owner, as well as biometric data in the form of a fingerprint and a photo. While this biometric data is not widely used at the moment, it is anticipated that the digitized photo will allow to automatize identity checks at border crossings or even perform cross-matching against lists of terrorism suspects (for a recent Interpol initiative to use face recognition to mass-screen passengers see [5]).
- The increasing deployment of surveillance cameras in public places (e.g. [18] estimates that 4.2 million surveillance cameras monitor the public in the UK) sparked interest in the use of face recognition technologies to automatically match faces of people shown on surveillance images against a database of known suspects. Despite massive technical problems that render this application currently infeasible, automatic biometric face recognition systems are still high on the agenda of policy makers [25, 19].

The ubiquitous use of face biometrics raises important privacy concerns; particularly problematic are scenarios where a face image is automatically matched against a database without the explicit consent of a person (for example in the above-mentioned surveillance scenario), as this allows to trace people against their will. The widespread use of biometrics calls for a careful policy, specifying to which party biometric data is revealed, in particular if biometric matching is performed at a central server or in partly untrusted environments.

In this paper we propose for the first time strong cryptographic Privacy-Enhancing Technologies for biometric face recognition; the techniques allow to hide the biometric data as well as the authentication result from the server that performs the matching. The proposed scheme can thus assure the privacy of individuals in scenarios where face recognition is beneficial for society but too privacy intrusive.

In particular, we provide a solution to the following two-party problem. Alice and Bob want to privately execute a standard biometric face recognition algorithm. Alice owns a face image, whereas Bob owns a database containing a collection of face images (or corresponding feature vectors) from individuals. Alice and Bob want to jointly run a face recognition algorithm in order to determine whether the picture owned by Alice shows a person whose biometric data is in Bob's database. While Bob accepts that Alice might learn basic parameters of the face recognition system (including the size of the database), he considers the content of his database as private data that he is not willing to reveal. In contrast, Alice trusts Bob to execute the algorithm correctly, but is neither willing to share the image nor the detection result with Bob. After termination, Alice will only learn if a match occurred; alternatively, an ID of the identified person may be returned.

In a real world scenario Bob might be a police organization, whereas Alice could be some private organization running an airport or a train station. While it may be common interest to use face recognition to identify certain people, it is generally considered too privacy intrusive to use Bob's central server directly for identification, as this allows him to create profiles of travelers. Thus, the two parties may decide for a privacy-friendly version where the detection result is not available to the central party. As the reputation of both parties is high and because both parties are interested in computing a correct result, it is reasonable to assume that they will behave in a semi-honest manner.

We provide a complete implementation of the above-mentioned two-party problem using the standard Eigenface [34] recognition system, working on encrypted images. At the heart of our privacy-enhanced face recognition system lies a highly optimized cryptographic protocol for comparing two Pailler-encrypted values. The system is very efficient and allows matching of an encrypted face image of size 92×112 pixels against a database of 320 facial templates in approximately 40 seconds on a conventional workstation. This is achieved despite the huge computational complexity of the underlying cryptographic primitives. Using pre-computations for intermediate values which do not depend on the input image, recognition only takes 18 seconds. While there is a small constant overhead when performing a face-recognition, the time to perform the recognition is linear in the size of the database. For a large database containing M facial templates, time for one recognition increases only slowly and requieres approximately $0.054M$ seconds for the conventional approach and $0.031M$ seconds when using pre-computations.

3.2 Cryptographic Tools

As a central cryptographic tool, we use two semantically secure additively homomorphic public-key encryption schemes, namely the Paillier and the DGK cryptosystem. In an additively homomorphic cryptosystem, given encryptions $[a]$ and $[b]$, an encryption $[a + b]$ can be computed by $[a + b] = [a][b]$, where all operations are performed in the algebra of the message or ciphertext space. Furthermore, messages can be multiplied with constants under encryption, i.e., given an encrypted message $[a]$ and a constant b in the clear, it is possible to compute $[ab]$ by $[ab] = [a]^b$.

Paillier cryptosystem. Introduced by Paillier in [29], its security is based on the decisional composite residuosity problem. Let $n = pq$ of size t, with p, q prime numbers and t from the range 1000-2048. Also let $g = n + 1$ [10]. To encrypt a message $m \in \mathbb{Z}_n$, the user selects a random value $r \in \mathbb{Z}_n$ and computes the ciphertext $c = g^m r^n \bmod n^2$. Note that due to our choice of g, encryption requires only one modular exponentiation and two modular multiplications, as $c = (mn+1)r^n \bmod n^2$. We will write the encryption of a message m in the Paillier cryptosystem as $[m]$. Since all encryptions in the proposed protocol will be computed using one fixed public key, we do not specify the key explicitly. It is easy to see that Paillier is additively homomorphic and that for an encryption $[m]$ we can compute a new probabilistic encryption of m without knowing the private key (this will be referred to as *re-randomization*). We refer the reader to [29] for a description of the decryption operation and further details on the cryptosystem.

Damgård, Geisler and Krøigaard cryptosystem (DGK). For efficiency reasons we use at a key point in our protocol another homomorphic cryptosystem, which was proposed by Damgård, Geisler and Krøigaard [8, 9]. As in Paillier, let $n = pq$ be a t-bit integer (with t chosen from the range 1000-2048), with p, q primes. The ciphertext c corresponding to a message $m \in \mathbb{Z}_u$ is computed as $c = g^m h^r \bmod n$, where u is a prime number and r is a randomly chosen integer. In practice (and more importantly in our application) u is from a very small range, say 8-bit values, which results in a very small plaintext space $\mathbb{Z}_u$. Similarly to Paillier, DGK is also additively homomorphic and it is possible to re-randomize existing ciphertexts. Compared to Paillier, the scheme has substantially smaller ciphertexts and the smaller plaintext space results in a large performance gain. To note the difference between Paillier and DGK ciphertexts we will denote the encryption of m in the DGK cryptosystem as $[\![m]\!]$.

3.3 Face Recognition

In 1991, Matthew Turk and Alex Pentland proposed an efficient approach to identify human faces [34, 35]. This approach transforms face images into characteristic feature vectors of a low-dimensional vector space (the face space), whose basis is composed of *eigenfaces*. The eigenfaces are determined through Principal Component Analysis (PCA) from a set of training images; every face image is succinctly represented as a vector in the face space by projecting the face image onto the subspace spanned by the eigenfaces. Recognition of a face is done by first projecting the face image to the face space and subsequently locating the closest feature vector. A more detailed description of the enrollment and recognition processes is given below.

During enrollment, a set of M training images $\Theta_1, \Theta_2, \ldots, \Theta_M$, which can be represented as vectors of length N, is used to determine the optimal low-dimensional face space, in which face images will be represented as points. To do this, the average of the training images is first computed as $\Psi = \frac{1}{M}\sum_{i=1}^{M}\Theta_i$. Then, this average is subtracted from each face vector to form difference vectors $\Phi_i = \Theta_i - \Psi$. Next, PCA is applied to the covariance matrix of these vectors $C = \frac{1}{M}\sum_{i=1}^{M}\Phi_i\Phi_i^T = \frac{1}{M}AA^T$ to obtain orthonormal eigenvectors and corresponding eigenvalues where A is the matrix where each column corresponds to the image Θ_i for $i = 1$ to M. (As the size of C makes it computationally infeasible to directly run PCA, the eigenvectors are usually obtained by applying PCA to the much smaller matrix A^TA and appropriate post-processing). At most M of the eigenvalues will be nonzero. To determine the face space, we select $T \ll M$ eigenvectors $u_1, \ldots, u_T$ that correspond to the T largest eigenvalues. Subsequently, images $\Theta_1, \Theta_2, \ldots, \Theta_M$ showing faces to be recognized (not necessarily the training images) are projected onto the subspace spanned by the basis $u_1, \ldots, u_T$ to obtain their feature vector representation $\Omega_1, \ldots, \Omega_M$.

During recognition, a new face image Γ is projected onto the face space by calculating weights $\bar{\omega}_i = u_i^T(\Gamma - \Psi)$ for $i = 1, \ldots, T$. These weights form a feature vector $\bar{\Omega} = (\bar{\omega}_1, \bar{\omega}_2, \ldots, \bar{\omega}_T)^T$ that represents the new image in the face space. Subsequently, the distances between the obtained vector $\bar{\Omega}$ and all feature vectors $\Omega_1, \ldots, \Omega_M$ present in the database are computed,

$$D_i = \|(\bar{\Omega} - \Omega_i)\|.$$

A match is reported if the smallest distance $D_{min} = \min\{D_1, \ldots, D_M\}$ is smaller than a given threshold value δ. Note that this basic recognition algorithm can be augmented with additional checks that reduce the number of false positives and negatives during recognition; for the sake of simplicity, we stick to the basic Eigenface recognition algorithm presented above.

3.4 Privacy-Preserving Eigenfaces

In this section, we present a privacy preserving realization of the Eigenface recognition algorithm which operates on encrypted images. We work in the two-party setting in the semi-honest attacker model. Informally, this assumes that the parties involved in the protocol follow it properly but keep a log of all the messages that have been exchanged (including their own) and try to learn as much information as possible from them. Alice's privacy is ensured against a computationally bounded attacker, while Bob's is unconditional—even a computationally unbounded Alice cannot compromise it. It is also assumed that the parties communicate over an authenticated channel (this can be achieved by standard mechanisms and is thus outside the scope of this paper).

3.4.1 Setup and Key Generation

Two parties Alice and Bob jointly run the recognition algorithm. We assume that Bob has already set up the face recognition system by running the enrollment process (in the clear) on all available training images to obtain the basis $u_1, \ldots, u_T$ of the face space and feature vectors $\Omega_1, \ldots, \Omega_M$ of faces to be recognized. Furthermore, we assume that all coordinates of the eigenfaces and feature vectors are represented as integers; this can always be achieved by appropriate quantization: non-integer values are first scaled by a fixed scale factor S and rounded to the nearest integer. This is necessary, as all values need to be integers in order to encrypt them with Paillier and process them using homomorphic operations. The effects of this quantization step on the detection reliability are experimentally analyzed in Section 3.7. Each feature vector in the database is further accompanied by a string Id_i that contains the identity of the person the feature vector belongs to; we assume that the identity is encoded as a *non-zero* element of the message space of the chosen encryption scheme.

During the interactive recognition protocol, Alice provides an encrypted face image $[\Gamma]$ as input. At the end of the protocol, Alice learns whether the face shown on her image matches one of the feature vectors $\Omega_1, \ldots, \Omega_M$ owned by Bob: Depending on the application, Alice either receives the identity Id_i of the best matching feature vector or only a binary answer (i.e. whether there was a match or not). Apart from this answer (and the number M), Bob keeps the database content secret. Bob learns nothing from the interaction, i.e. neither the face image Γ, nor its representation in the face space, nor the result of the matching process.

Note that the vectors u_i are directly computed from the set of training images; thus, they *do* carry information on the faces stored in Bob's database. Even though it is hard to quantify the exact amount of data leakage through the knowledge of the basis $u_1, \ldots, u_T$, our solution will treat it as sensitive data that will not be disclosed to Alice. In an alternative implementation, the basis $u_1, \ldots, u_T$ can be derived from a sufficiently large public face database so that they do not carry personal information; the proposed system can easily be changed to take advantage of public basis vectors, see Section 3.7 for details. Since Alice is the only party who receives an output, we can construct the protocol using any standard homomorphic public-key encryption algorithm; as stated in Section 3.2 we choose Paillier encryption for the implementation. In particular, we do *not* need a threshold homomorphic scheme, as it is widely employed in the construction of secure multiparty protocols. Before the interaction starts, Alice generates a pair of public and private keys and sends her public key to Bob over an authenticated channel. In the first step of the protocol, Alice encrypts all pixels of the image Γ separately with her public key and sends the result to Bob, who is unable to decrypt them. However, Bob can use the homomorphic property of the cipher to perform linear operations on the ciphertexts; for some operations (such as computing distances between vectors or finding a minumum), he will require assistance from Alice in the form of an interactive protocol. At the end of the protocol, Alice receives back an encryption containing the result of the biometric matching operation, which only Alice can decrypt. Appendix 3.6 gives a sketch of the security of our system in the semi-honest attacker model.

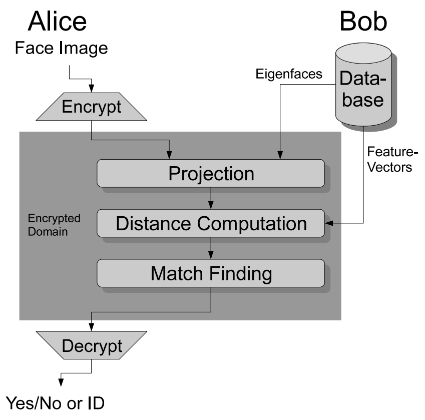


Figure 3.1: Privacy-Preserving Face Recognition.

3.4.2 Private Recognition Algorithm

To match a face image against feature vectors in a database, three steps need to be performed. First, the image needs to be projected onto the face space in order to obtain its corresponding feature vector representation. Subsequently, distances between the obtained vector and all feature vectors in Bob's database need to be computed. Finally, the one with minimum distance is selected; if this distance is smaller than a threshold, a match is reported. In the following, we show how these three steps can be realized in a privacy preserving manner. Figure 3.1 shows an outline of the private face recognition protocol; the gray area denotes operations that need to be performed on encrypted values.

Projection

As a first step, the input image Γ has to be projected onto the low dimensional face space spanned by the eigenfaces $u_1, \ldots, u_T$. This can be performed by computing the scalar product of

$$\Phi = \Gamma - \Psi = \begin{pmatrix} \Gamma_1 - \Psi_1 \\ \vdots \\ \Gamma_N - \Psi_N \end{pmatrix}$$

and each eigenface vector u_i to obtain

$$\bar{\omega}_i = \Phi_1 \cdot u_{i1} + \ldots + \Phi_N \cdot u_{iN}$$

for each $i \in \{1, \ldots, T\}$.

These operations have to be performed in the encrypted domain by Bob, who receives the encrypted face image $[\Gamma]$ from Alice. As Bob knows the vector Ψ in plain, he can easily compute $-\Psi = (-1) \cdot \Psi$ and then encrypt each of its components. These encryptions can be pairwise multiplied with the encrypted components of $[\Gamma]$

in order to perform the componentwise subtraction of the vectors Γ and Ψ. Thus Bob computes

$$[\Phi] = [\Gamma - \Psi] = \begin{pmatrix} [\Gamma_1] \cdot [-\Psi_1] \\ \vdots \\ [\Gamma_N] \cdot [-\Psi_N] \end{pmatrix}.$$

Subsequently Bob performs the projection

$$[\bar{\omega}_i] = [\Phi_1 \cdot u_{i1} + \ldots + \Phi_N \cdot u_{iN}] = [\Phi_1]^{u_{i1}} \cdot \ldots \cdot [\Phi_N]^{u_{iN}}$$

for each $i \in \{1, \ldots, T\}$. This is done as follows. As Bob knows the vector u_i in plain, he can perform the required multiplications using the homomorphic property. For example, in order to multiply the first components of both vectors Bob has to compute $[\Phi_1]^{u_{i1}}$. To obtain the sum of all these products he just multiplies the encryptions with each other. Doing this for all $1 \leq i \leq T$, Bob obtains an encrypted feature vector description of the face image as $[\bar{\Omega}] := ([\bar{\omega}_1], \ldots, [\bar{\omega}_T])^T$. Note that every computation in the projection operation can be performed by Bob without interacting with Alice.

Calculating Distances

After having obtained the encrypted feature vector $[\bar{\Omega}]$, encryptions of the distances $D_1, \ldots, D_M$ between $\bar{\Omega}$ and all feature vectors $\Omega \in \{\Omega_1, \ldots, \Omega_M\}$ from the database have to be computed. Since in the remainder of the protocol we are only concerned with the relative order of the obtained distances, it suffices to compute the square of the Euclidean distance,

$$\begin{aligned} D(\Omega, \bar{\Omega}) &= \|\Omega - \bar{\Omega}\|^2 = (\omega_1 - \bar{\omega}_1)^2 + \ldots + (\omega_T - \bar{\omega}_T)^2 \\ &= \underbrace{\sum_{i=1}^{T} \omega_i^2}_{\mathcal{S}_1} + \underbrace{\sum_{i=1}^{T} (-2\omega_i \bar{\omega}_i)}_{\mathcal{S}_2} + \underbrace{\sum_{i=1}^{T} \bar{\omega}_i^2}_{\mathcal{S}_3}. \end{aligned} \tag{3.1}$$

Again, we need to evaluate this equation in the encrypted domain: Bob knows the encryption $[\bar{\Omega}]$ and needs to compute the encrypted distance $[D(\Omega, \bar{\Omega})]$, while he knows the feature vector Ω in the clear. To compute $[D(\Omega, \bar{\Omega})]$ it suffices to compute encryptions of the three sums $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$, as by the homomorphic property and Eq. (3.1),

$$[D(\Omega, \bar{\Omega})] = [\mathcal{S}_1] \cdot [\mathcal{S}_2] \cdot [\mathcal{S}_3].$$

The term $\mathcal{S}_1$ is the sum over the components of Ω known in the clear. Thus, Bob can compute $\mathcal{S}_1$ directly and encrypt it to obtain $[\mathcal{S}_1]$. $\mathcal{S}_2$ consists of the products $\omega_i \bar{\omega}_i$, where Bob knows ω_i in the clear and has $[\bar{\omega}_i]$ in encrypted form. In a first step the values ω_i can be multiplied with -2. The term $[(-2\omega_i)\bar{\omega}_i]$ can be computed by raising $[\bar{\omega}_i]$ to the power of $(-2\omega_i)$, using the homomorphic property. To obtain an encryption of $\mathcal{S}_2$, Bob finally computes $[\mathcal{S}_2] = \prod_{j=1}^{T} [(-2\omega_i)\bar{\omega}_i]$. Thus, the value $[\mathcal{S}_2]$ can again be computed by Bob without interacting with Alice. The term $\mathcal{S}_3$ consists of the squares of the encrypted values $[\bar{\omega}_i]$. Unfortunately, Bob cannot perform the required multiplication without help from Alice. Thus, Bob additively blinds the value $\bar{\omega}_i$ with an uniformly random element r_i from the plaintext space to obtain $[x_i] = [\bar{\omega}_i + r_i] = [\bar{\omega}_i] \cdot [r_i]$. Note that for every component $\bar{\omega}_i$ of the vector $\bar{\Omega}$ a fresh random value must be generated. Finally, he sends the elements $[x_i]$ to Alice who decrypts. Alice can now compute the values x_i^2 in plain as the square of the plaintext x_i and compute the value $\mathcal{S}'_3 = \sum_{j=1}^{T} x_i^2$. She encrypts this value and sends $[\mathcal{S}'_3]$ back to Bob, who computes

$$[\mathcal{S}_3] = [\mathcal{S}'_3] \cdot \prod_{j=1}^{T} ([\bar{\omega}_i]^{(-2r_i)} \cdot [-r_i^2]),$$

which yields the desired result because

$$[x_i^2] \cdot [\bar{\omega}_i]^{(-2r_i)} \cdot [-r_i^2] = [(\bar{\omega}_i + r_i)^2 - 2r_i\bar{\omega}_i - r_i^2] = [\bar{\omega}_i^2].$$

Note that this interactive protocol to compute the value $[\mathcal{S}_3]$ needs to be run only once. The value $[\mathcal{S}_3]$ depends only on the encrypted feature vector $[\bar{\Omega}]$ and can be used for computation of all distances $[D_1], \ldots, [D_M]$. Note further that due to the blinding factors, Alice does not learn the values $\bar{\omega}_i$.

Match Finding

In the last step of the recognition algorithm, the feature vector from the database that is closest to $\bar{\Omega}$ must be found. This distance is finally compared to a threshold value δ; if the distance is smaller, a match is reported and an encryption of the identity Id which corresponds to the best matching feature vector is returned to Alice.

As a result of the last step we obtained encrypted distances $[D_1], \ldots, [D_M]$, where D_i denotes the distance between $\bar{\Omega}$ and the i-th feature vector $\Omega_i \in \{\Omega_1, \ldots, \Omega_M\}$ from the database. To find the minimum we employ a straightforward recursive procedure: in the first step, we compare the $k = \lfloor \frac{M}{2} \rfloor$ encrypted distances $[D_{2i+1}]$ and $[D_{2i+2}]$ for $0 \leq i \leq k-1$ with each other, by using a cryptographic protocol that compares two encrypted values; a re-randomized encryption of the smaller distance is retained (re-randomization is necessary to prevent Bob from determining the outcome of the comparison by inspecting the ciphertexts). After this step, there will be $\lceil \frac{M}{2} \rceil$ encryptions left. In a second run we repeat this procedure for the remaining encryptions, and so forth. After $\lceil \log_2(M) \rceil$ iterations there will only be one encryption left, the minimum.

As we need to return the identity of the best matching feature vector, we also have to keep track of the IDs during the minimum computation. This is done by working with pairs $([D_i], [Id_i])$ of distances and their corresponding identities, where the recursive minimum finding algorithm is applied to the distances only, but re-randomized encryptions of both the smaller distance and its identity are retained for the next round. An efficient implementation of the required comparison protocol is described in Section 3.5.

To check if the minimum distance is smaller than a threshold δ, we can treat the value δ as one additional distance that has the special identity 0. Together with the distances $D_1, \ldots, D_M$ we run the algorithm to find the minimum as described above. After $\lceil \log_2(M+1) \rceil$ iterations, Bob receives the minimum distance and the corresponding identity $([D], [Id])$, where $D \in \{\delta, D_1, \ldots, D_M\}$ and $Id \in \{0, Id_1, \ldots, Id_M\}$. Thus, if a face image could be recognized the value Id contains the corresponding identity. If no match could be found Id is equal to 0. The value $[Id]$ is finally sent to Alice as the result of the private face recognition protocol.

Note that there is an easy way to modify the protocol to make it terminate only with a binary output: rather than using actual IDs, Bob may assign a second special identity, the integer 1, to all images. In this case Alice will either receive a 1 or a 0, with the former indicating that a match was found.

3.5 Comparison Protocol

The only missing block is a protocol for selecting the minimum of two encrypted ℓ-bit values $[a]$ and $[b]$ along with the encrypted ID of the minimum. (Note that the bit-length ℓ can be determined by knowing the bit-length of the input data and the scale factor S used to quantize eigenfaces).

At the core of our protocol is a comparison protocol due to Damgård, Geisler and Krøigaard [8, 9]. Their setting differs from ours as follows: one input is public while the other is held (bitwise) in encrypted form by one party; moreover the output is public. They note several variations, but in order to provide a solution for the present setting some tweaking is needed. This section presents the protocol in a top-down fashion.

3.5.1 A High-level View of the Protocol

Initially Bob, who has access to both $[a]$ and $[b]$, computes

$$[z] = [2^\ell + a - b] = [2^\ell] \cdot [a] \cdot [b]^{-1}.$$

As $0 \leq a, b < 2^\ell$, z is a positive $(\ell+1)$-bit value. Moreover, z_ℓ, the most significant bit of z, is exactly the answer we are looking for:

$$z_\ell = 0 \Leftrightarrow a < b.$$

If Bob had an encryption of $z \bmod 2^\ell$, the result would be immediate: z_ℓ could be computed as

$$z_\ell = 2^{-\ell} \cdot (z - (z \bmod 2^\ell)).$$

Correctness is easily verified; the subtraction sets the least significant bits to zero, while the multiplication shifts the interesting bit down. As only z and $z \bmod 2^\ell$ are encrypted, this is a linear combination in the encrypted domain, which can be computed by Bob.

Once Bob has an encryption of the outcome $[z_\ell] = [a < b]$, an encryption of the minimum m, is easily obtained using arithmetic, as $m = (a < b) \cdot (a - b) + b$. The multiplication requires assistance of Alice, but is easily performed through a (short) interactive protocol. Determining an encryption of the ID is analogous, $(a < b) \cdot$

$(Id_a - Id_b) + Id_b$. Thus, it remains to describe how Bob obtains the encryption of $z \bmod 2^\ell$.

3.5.2 Computing $[z \bmod 2^\ell]$

The value z is available to Bob only in encrypted form, so the modulo reduction cannot easily be performed. The solution is to engage in a protocol with Alice, transforming the problem back to a comparison.

First, Bob generates a uniformly random $(\kappa + \ell + 1)$-bit value r, where κ is a security parameter, say 100, and $\kappa + \ell + 1 \ll \log_2(n)$. This will be used to additively blind z,

$$[d] = [z + r] = [z] \cdot [r];$$

$[d]$ is then re-randomized and sent to Alice who decrypts it and reduces d modulo 2^ℓ. The obtained value is then encrypted, and returned to Bob.

Due to the restriction on the bit-length of r, Bob can now *almost* compute the desired encryption $[z \bmod 2^\ell]$. The masking can be viewed as occurring over the integers, thus we have $d \equiv z + r \bmod 2^\ell$ and

$$\big(z \bmod 2^\ell\big) = \Big(\big(d \bmod 2^\ell\big) - \big(r \bmod 2^\ell\big)\Big) \bmod 2^\ell.$$

Alice has just provided $[d \bmod 2^\ell]$ and r is known to Bob. Thus, he can compute

$$[\tilde{z}] = [(d \bmod 2^\ell) - (r \bmod 2^\ell)] = [d \bmod 2^\ell] \cdot [(r \bmod 2^\ell)]^{-1}.$$

Had the secure subtraction occurred modulo 2^ℓ, $\tilde{z}$ would be the right result; however, it occurs modulo n. Note, though, that if $d \bmod 2^\ell \geq r \bmod 2^\ell$, $\tilde{z}$ is the right result. On the other hand, if $r \bmod 2^\ell$ is larger, an underflow has occurred; adding 2^ℓ in this case gives the right result. So, if Bob had an encryption $[\lambda]$ of a binary value indicating whether $r \bmod 2^\ell > d \bmod 2^\ell$, he could simply compute

$$[z \bmod 2^\ell] = [\tilde{z} + \lambda 2^\ell] = [\tilde{z}] \cdot [\lambda]^{2^\ell},$$

which adds 2^ℓ exactly when $r \bmod 2^\ell$ is the larger value. This leaves us with a variant of Yao's millionaires problem: Bob must obtain an encryption $[\lambda]$ of a binary value containing the result of the comparison of two private inputs: $\hat{d} = d \bmod 2^\ell$ held by Alice and $\hat{r} = r \bmod 2^\ell$ held by Bob.

3.5.3 Comparing Private Inputs

The problem of comparing private inputs $\hat{d}$ and $\hat{r}$ is a fundamental one, which has been studied intensively (see e.g. [38, 28, 14, 3, 4, 15, 8]). For efficiency reasons, we solve this problem using a *different* homomorphic encryption scheme, namely the one proposed by Damgård et al. [8, 9], which has a very small plaintext space $\mathbb{Z}_u$ for some prime u. This allows very efficient multiplicative masking; in contrast to the Paillier scheme, the exponents are small.

Though the basic setting of Damgård et al. considers one public and one secret value, they note how to construct a solution for private inputs. They also note how to obtain a secret output. However, they obtain this output as an additive secret sharing, while in our setting Bob must receive a *Paillier encryption* $[\lambda]$ at the end of the protocol. Naturally Alice must not see this encryption as she knows the secret key.

We assume that Alice has run the DGK key-generation algorithm and has sent the public key to Bob. This key pair can be re-used whenever the comparison protocol will be run. Inertially, Alice sends Bob encryptions of the bits of her input, $[\![\hat{d}_{\ell-1}]\!], \ldots, [\![\hat{d}_0]\!]$. Bob then chooses $s \in_R \{1, -1\}$ and computes

$$[\![c_i]\!] = [\![\hat{d}_i - \hat{r}_i + s + 3 \sum_{j=i+1}^{\ell-1} w_j]\!] = [\![\hat{d}_i]\!] \cdot [\![-\hat{r}_i]\!] \cdot [\![s]\!] \cdot \left(\prod_{j=i+1}^{\ell-1} [\![w_j]\!] \right)^3 , \qquad (3.2)$$

where $[\![w_j]\!] = [\![\hat{d}_j \oplus \hat{r}_j]\!]$, which he can compute as Bob knows $\hat{r}_j$. For technical reasons (to avoid the case $\hat{d} = \hat{r}$), we append differing bits to both $\hat{d}$ and $\hat{r}$, i.e., we compare the values $2\hat{d} + 1$ and $2\hat{r}$ instead.

Equation (3.2) differs from the one proposed by Damgård et al. in order to efficiently hide the output, but the core idea remains. Consider the case of $s = 1$; if $\hat{d}$ is larger, then all c_i will be non-zero. (The modulus u is chosen such that there is no overflow.) However, if $\hat{r}$ is larger, then exactly one c_i will equal zero, the one at the most significant differing bit-position. Both claims are easily verified. For $s = -1$ we have exactly the same situation, except that the zero occurs if $\hat{d}$ is larger. The factor of 3 ensures that the values are non-zero once even a single w_j is set.

Bob now multiplicatively masks the $[\![c_i]\!]$ with a uniformly random $r_i \in \mathbb{Z}_u^*$

$$[\![e_i]\!] = [\![c_i \cdot r_i]\!] = [\![c_i]\!]^{r_i},$$

re-randomizes and permutes the encryptions $[\![e_i]\!]$ and sends them to Alice. Note that e_i is uniformly random in $\mathbb{Z}_u^*$ except when $c_i = 0$, in which case e_i also equals zero, i.e. the existence of a zero is preserved.

Alice now decrypts all e_i and checks whether one of them is zero. She then encrypts a bit $\tilde{\lambda}$, stating if this is the case. At this point she switches back to Paillier encryptions, i.e. Alice sends $[\tilde{\lambda}]$ to Bob. Given the knowledge of s, Bob can compute the desired encryption $[\lambda]$: while $[\tilde{\lambda}]$ only states whether there was a zero among the values decrypted by Alice, s explains how to interpret the result, i.e. whether the occurrence of a zero means that $\hat{r} > \hat{d}$ or $\hat{d} \geq \hat{r}$. In the former case, Bob negates the result $[\tilde{\lambda}]$ under encryption, otherwise he directly takes $[\tilde{\lambda}]$ as output $[\lambda]$.

3.6 Security (Sketch)

In this appendix we sketch why the face recognition protocol is privacy preserving. For semi-honest Alice and Bob, neither learns anything on the other's input—the database and the image—except the database size and what can be inferred from the output. As the parties are honest-but-curious, it suffices to demonstrate that no information is leaked by the messages seen.

Comparison protocol. The comparison protocol allows Bob to obtain a new encryption of the minimum of two encryptions he already possesses. On the intuitive level, security towards Bob is simple. All messages received are encrypted under Alice's public keys, and Bob cannot learn anything from these without breaking the semantic security of one of those schemes.

Alice on the other hand has access to the secret key. It must therefore be argued that no information is learned from the *contents* of the encryptions sent. But this is the case, as Alice only receives values that Bob has masked: this includes the messages sent for the secure selection of the minimal and ID, as well as $[d] = [z + r]$, which is statistically indistinguishable from a uniformly random $(\kappa + \ell + 1)$-bit value.

Treatment of the permuted $[\![e_i]\!]$ of Section 3.5.3 is only slightly more difficult. Alice either sees a list of uniformly random non-zero values, or an equivalent list, where one entry is replaced by a zero. A list of random values provides no information. Similarly, the zero does not cause any problems: Its position is random due to the permutation, and its existence also reveals nothing as it occurs with probability $1/2$; s can be viewed as a one-time-pad for the outcome. Thus, neither Alice nor Bob learn anything from the comparison protocol.

Complete Recognition Protocol. The proof of security of the full protocol is similar to that of the comparison. In addition to the comparisons, interaction is only needed to compute the distances $D_1, \ldots, D_M$. As above, the values $x_1, \ldots, x_T$ that Alice receives are masked, in this case they are uniformly random over the whole plaintext space. Bob again receives only semantically secure encryptions, so he also learns nothing. This is also true when he receives Alice's input.

Based on the above intuition, a formal simulator proof is easily constructed. Given one party's input and the output, simulation of the other party is easy: Alice must be handed encryptions of random values, while Bob can be handed encryptions of 0, which are indistinguishable due to the semantic security.

3.7 Implementation

The privacy-preserving face recognition system, as described in this paper, has been implemented in C++ using the GNU GMP library version 4.2.4, in order to determine its performance and reliability. Tests were performed on a computer with a 2.4 GHz AMD Opteron dual-core processor and 4GB of RAM running Linux. Both sender and receiver were modeled as different threads of one program, which pass messages to each other; thus, the reported performance data does not include network latency.

For testing purposes, we used the "ORL Database of Faces" from AT&T Laboratories Cambridge [1], which is widely used for experiments and contains 10 images of 40 distinct subjects, thus 400 images in total. All images in this database have a dark background with the subject in upright, frontal position. The size of each image is 92×112 pixels with 256 grey levels per pixel (thus $N = 92 \cdot 112 = 10304$). We use 5-fold cross validation for the experiments such that for each subject we use 8 images in the enrollment phase and 2 images for testing (thus, the database consists of 320 feature vectors). The security parameter k for both Paillier- and DGK-cryptosystem was set to 1024 bits (see Section 3.2 for details). Furthermore we set $\ell = 50$ (see Section 3.5 for details).

Reliability. During reliability testing, we assured that our privacy-preserving implementation of the Eigenface algorithm does not degrade the reliability when compared to a standard implementation which achieves approximately 96% correct classification rate. Reliability losses may occur due to the use of scaled and quantized feature vectors and eigenfaces. This scale factor has both an influence on the accuracy of the result and the performance of the scheme. Figure 3.2 shows the detection rates of the implementation for different scale factors, plotted on a logarithmic scale. It can be seen that scale factors below the value 1000 significantly degrade detection performance, while scale factors larger than 1000 do not improve the results. Hence, it suffices to set $S = 1000$ to achieve the same reliability as a reference implementation operating on floating point values. Another parameter that influences both the detection rate and the performance is the number T. Turk and Pentland [34] advised to set $T = 10$; experiments with our implementation demonstrate that values of $T > 12$ do not yield a significant gain in the detection rate; thus we set $T = 12$ in subsequent tests.

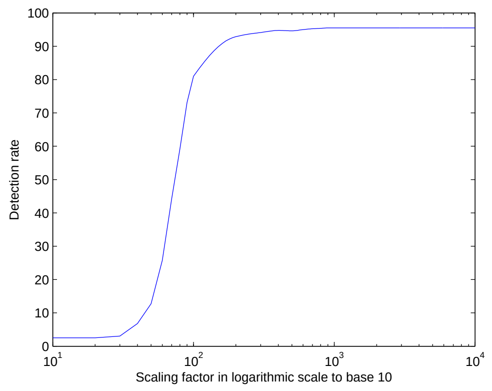


Figure 3.2: Relation between scale factor and detection rate.

Computational complexity. We measure the computational complexity of the full recognition protocol, thus the efforts of both Alice and Bob. Table 3.1 depicts the average runtime of a single query (wall clock time) with respect to the size of the database M (second column) in seconds. Thus, matching an image against a database of size 320 takes roughly 40 seconds; this time includes all steps of the protocol of Section 3.4: computing the encrypted face image by Alice, projecting it into the face space, computing distances and selecting the minimum.

One can note that a major part of the computation efforts comes from computing encryptions, since they require one rather complex modular exponentiation. The time required to run the protocol can be largely reduced if these computationally expensive operations, which do *not* depend on the input image of Alice, can be computed in advance, during idle times of a processor or on a separate processor dedicated to this task. With this optimization in place, computing one encryption requires only two modular multiplications. The third column of Table 3.1 shows the execution time of the recognition algorithm under the assumption that *all* randomization factors r^n (Paillier) and h^r (DGK) can be pre-computed for free during idle times. In this case, matching an image against 320 feature vectors takes only 18 seconds; furthermore, the computations performed by Alice become much more lightweight, as nearly all of Alice's efforts is spent in computing encryptions.

In a third test we assume that Alice knows the eigenfaces u_i. As noted in Section 3.4.1, this might be the case if a (sufficiently large) public database of faces can be used to compute the eigenfaces, or if Bob explicitly decides to reveal these values to Alice. In this case Alice performs the projection and distance computation steps and sends an encrypted feature vector to Bob. The results of this experiment are depicted in the fourth column of Table 3.1. Observe that compared to a standard query (second column) only a small constant factor can be saved.

Communication complexity. The communication complexity highly depends on the size of Paillier and DGK encryptions; in our implementation, the size of a Paillier ciphertext is 2048 bits, whereas a DGK encryption requires only 1024 bits. Sending the encrypted image and performing the distance computations requires communication efforts independent of M; in particular, this part of the protocol requires transmission of $N + T + 1$ Paillier encrypted values (roughly 2580 kilobytes). The rest of the communication is linear in M: more precisely, the minimum searching step requires transmission of $6M$ Pailler and $M(2\ell + 1)$ DGK encryptions, which in our setting amounts to roughly 14.5 kilobytes per feature vector in the database. Table 3.2 shows the average amount of data in kilobytes transmitted in one run of the privacy-preserving face recognition protocol for several database sizes M (second column) and the communication complexity in case that a public basis of Eigenfaces can be used (third column). The overall communication complexity for matching an image against 320 feature vectors is thus approximately 7.25 MB.

Table 3.1: Computational Complexity (sec.).

M	Query	With pre-computations	Public Eigenfaces
10	24	8.5	1.6
50	26	10	3.4
100	29	11.5	6
150	31.6	13	8.6
200	34.2	14.5	11.4
250	36.6	16	14.4
300	39.6	17.5	18
320	40	18	18.2

Round complexity. The round complexity of our protocol is very low. Sending the face image and receiving the result of the protocol takes one round. Another round is spent for distance computation. As the comparison protocol (see Section 3.5) runs in three rounds, finding the minimum of $M + 1$ values takes at most $3\lceil\log_2(M + 1)\rceil$ rounds. Therefore the round complexity of our protocol is $\mathcal{O}(\log_2(M))$.

Table 3.2: Communication Complexity(kB).

M	Full Query	Public Eigenfaces
10	2725	149
50	3310	734
100	4038	1461
150	4765	2189
200	5497	2921
250	6228	3652
300	6959	4382
320	7249	4674

3.8 Related Work

The problem considered in this paper is an instance of a secure two-party problem; thus standard methods of Secure Multiparty Computation [38, 7] can be applied. Basic concepts for secure computations were introduced by Yao [38]. Subsequently, various approaches to securely evaluating a function have been developed for different function representations, namely combinatorial circuits [17, 20], ordered binary decision diagrams [24], branching programs [27, 26], or one-dimensional look-up tables [26]. Nevertheless, these solutions tend to be impractical due to their high computational complexity for functions as the biometric matching process considered in this paper. Thus, specific protocols must be developed.

Recently there has been an increasing interest in the use of SMC for data-intensive problems, like clustering [16, 21], filtering [6] or statistical analysis [11] of sensitive private data. Furthermore, the combination of signal processing with cryptographic techniques in order to protect privacy is an active area of research [13]; among others, solutions for recognizing speech on encrypted signals [33] or image classification and object recognition on encrypted images [37, 2] have been proposed. The latter work describes a solution to a problem that is complementary to the one discussed in the present paper (and can be used in conjunction with our solution): locating rectangular regions on an encrypted image that show human faces.

Some authors proposed different complementary techniques for making surveillance cameras more privacy friendly, e.g. [32, 12, 39]. However, they do not consider face recognition. These approaches use methods from signal processing and pattern recognition to wipe out sensitive regions of a surveillance video automatically, based on access permissions of the surveillance personnel.

There were a few attempts to make other biometric modalities privacy-preserving, most notably fingerprints and iris codes [36, 30, 23]. However, these works consider a different setting, where the biometric measurement is matched against a hashed template stored on a server. The server that performs the matching gets to know both the biometric and the detection result (the aim is only to secure storage of templates). In contrast, our scenario even allows to hide this information. There are only a few works that apply cryptographic secure multiparty computation to the problem of securing iris codes and fingerprint templates (most notably [22, 31]); to the best of our knowledge there is no prior solution to the much more data-intensive problem of securing face biometrics.

3.9 Conclusions and Future Work

In this paper we have presented for the first time strong cryptographic privacy enhancing technologies for biometric face recognition systems. In particular, we provided an efficient protocol that allows to match an encrypted image showing a face against a database of facial templates in such a way that the biometric itself and the detection result is hidden from the server that performs the matching. Through extensive tests, we showed that our privacy-preserving algorithm is as reliable as a reference implementation in the clear, and that the execution of the protocol is feasible on current hardware platforms.

In this paper we used Eigenfaces, which provides a detection rate of about 96%, as core face recognition algorithm. Biometric algorithms that achieve better detection rates are known in the literature; however, these schemes are much more complex and thus more difficult to implement on encrypted images. We leave this, as well as further optimizations, as future work.

References

[1] The Database of Faces, (formerly 'The ORL Database of Faces'). Available at http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html. AT&T Laboratories Cambridge.

[2] S. Avidan and M. Butman. Blind vision. In *Computer Vision - ECCV 2006, 9th European Conference on Computer Vision*, volume 3953 of *LNCS*, pages 1–13. Springer, 2006.

[3] I. F. Blake and V. Kolesnikov. Strong Conditional Oblivious Transfer and Computing on Intervals. In P. J. Lee, editor, *Advances in Cryptology — ASIACRYPT 2004*, volume 3329 of *LNCS*, pages 515–529. Springer, December 5-9, 2004.

[4] I. F. Blake and V. Kolesnikov. Conditional Encrypted Mapping and Comparing Encrypted Numbers. In G. D. Crescenzo and A. D. Rubin, editors, *Financial Cryptography and Data Security — FC 2006*, volume 4107 of *LNCS*, pages 206–220. Springer, February 27-March 2, 2006.

[5] O. Bowcott. Interpol wants facial recognition database to catch suspects. Guardian, October 20, 2008, http://www.guardian.co.uk/world/2008/oct/20/interpol-facial-recognition.

[6] J. F. Canny. Collaborative filtering with privacy. In *IEEE Symposium on Security and Privacy*, pages 45–57, 2002.

[7] R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty Computation from Threshold Homomorphic Encryption. In B. Pfitzmann, editor, *Advances in Cryptology — EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 280–299. Springer, May 6-10, 2001.

[8] I. Damgård, M. Geisler, and M. Krøigaard. Efficient and Secure Comparison for On-Line Auctions. In J. Pieprzyk, H. Ghodosi, and E. Dawson, editors, *Australasian Conference on Information Security and Privacy — ACSIP 2007*, volume 4586 of *LNCS*, pages 416–430. Springer, July 2-4, 2007.

[9] I. Damgård, M. Geisler, and M. Krøigaard. A correction to "Efficient and secure comparison for on-line auctions". Cryptology ePrint Archive, Report 2008/321, 2008. http://eprint.iacr.org/.

[10] I. Damgård and M. Jurik. A Generalization, a Simplification and some Applications of Paillier's Probabilistic Public-Key System. Technical report, Department of Computer Science, University of Aarhus, 2000.

[11] W. Du, Y. S. Han, and S. Chen. Privacy-preserving multivariate statistical analysis: Linear regression and classification. In *Proceedings of the Fourth SIAM International Conference on Data Mining, Lake Buena Vista, Florida, USA*, pages 222–233. SIAM, April 22-24, 2004.

[12] F. Dufaux and T. Ebrahimi. Scrambling for video surveillance with privacy. In *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*. IEEE Press, 2006.

[13] Z. Erkin, A. Piva, S. Katzenbeisser, and et al. Protection and retrieval of encrypted multimedia content: When cryptography meets signal processing. *EURASIP Journal on Information Security*, 2007, Article ID 78943.

[14] M. Fischlin. A Cost-Effective Pay-Per-Multiplication Comparison Method for Millionaires. In D. Naccache, editor, *Topics in Cryptology - CT-RSA 2001*, volume 2020 of *LNCS*, pages 457–472. Springer, April 8-12, 2001.

[15] J. A. Garay, B. Schoenmakers, and J. Villegas. Practical and Secure Solutions for Integer Comparison. In T. Okamoto and X. Wang, editors, *Public Key Cryptography — PK 2007*, volume 4450 of *LNCS*, pages 330–342. Springer, April 16-20, 2007.

[16] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen. On secure scalar product computation for privacy-preserving data mining. In *7th ICISC*, volume 3506 of *LNCS*, pages 104–120. Springer, 2004.

[17] O. Goldreich, S. Micali, and A. Wigderson. How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority. In *ACM Symposium on Theory of Computing — STOC '87*, pages 218–229. ACM, May 25-27, 1987.

[18] T. Grose. When surveillance cameras talk. Time Magazine, February 11, 2008, `http://www.time.com/time/world/article/0,8599,1711972,00.html`.

[19] Interpol wants facial recognition database to catch suspects. Heise Online UK, March 20, 2008, `http://www.heise-online.co.uk/news/British-police-build-a-database-of-portrait-photos-for-facial-recognition/110363`.

[20] M. Jacobsson and A. Juels. Mix and match: Secure function evaluation via ciphertexts. In T. Okamoto, editor, *Advances in Cryptology – ASIACRYPT'00*, volume 1976 of *LNCS*, pages 162–177. Springer-Verlag, 2000.

[21] G. Jagannathan and R. N. Wright. Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 593–599, New York, NY, USA, 2005. ACM Press.

[22] F. Kerschbaum, M. J. Atallah, D. M'Raïhi, and J. R. Rice. Private fingerprint verification without local storage. In *Biometric Authentication, First International Conference, ICBA 2004*, volume 3072 of *LNCS*, pages 387–394. Springer, 2004.

[23] T. Kevenaar. *Security with Noisy Data*, chapter Protection of Biometric Information, pages 169–193. Springer Verlag, 2007.

[24] L. Kruger, S. Jha, E.-J. Goh, and D. Boneh. Secure function evaluation with ordered binary decision diagrams. In *Proceedings of the 13th ACM conference on Computer and communications security CCS'06*, pages 410–420, Virginia, U.S.A., November 2006. ACM Press.

[25] M. Magnier. Many eyes will watch visitors. Los Angeles Times, August 07, 2008, `http://articles.latimes.com/2008/aug/07/world/fg-snoop7`.

[26] M. Naor and K. Nissim. Communication complexity and secure function evaluation. *Electronic Colloquium on Computational Complexity (ECCC)*, 8(062), 2001.

[27] M. Naor and K. Nissim. Communication preserving protocols for secure function evaluation. In *ACM Symposium on Theory of Computing*, pages 590–599, 2001.

[28] M. Naor, B. Pinkas, and R. Sumner. Privacy preserving auctions and mechanism design. In *ACM Conference on Electronic Commerce*, pages 129–139, 1999.

[29] P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In J. Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *LNCS*, pages 223–238. Springer, May 2-6, 1999.

[30] N. Ratha, J. Connell, R. Bolle, and S. Chikkerur. Cancelable biometrics: A case study in fingerprints. In *Proceedings of the 18th International Conference on Pattern Recognition (ICPR, volume IV)*, pages 370–373. IEEE Press, 2006.

[31] B. Schoenmakers and P. Tuyls. *Security with Noisy Data*, chapter Computationally Secure Authentication with Noisy Data, pages 141–149. Springer Verlag, 2007.

[32] A. Senior, O. A, and A. H. et al. Enabling video privacy through computer vision. *IEEE Security and Privacy Magazine*, 3(3):50–57, 2005.

[33] P. Smaragdis and M. Shashanka. A framwork for secure speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 15(4):1404–1413, 2007.

[34] M. A. Turk and A. P. Pentland. Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[35] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, pages 586–591, 1991.

[36] P. Tuyls, A. H. M. Akkermans, T. A. M. Kevenaar, G. J. Schrijen, A. M. Bazen, and R. N. J. Veldhuis. Practical biometric authentication with template protection. In *Audio- and Video-Based Biometric Person Authentication, 5th International Conference, AVBPA 2005*, volume 3546 of *LNCS*, pages 436–446. Springer, 2005.

[37] J. Vaidya and B. Tulpule. Enabling better medical image classification through secure collaboration. In *Proc. IEEE International Conference on Image Processing (ICIP)*, pages IV–461–IV–464, 2007.

[38] A. C.-C. Yao. Protocols for Secure Computations (Extended Abstract). In *Annual Symposium on Foundations of Computer Science — FOCS '82*, pages 160–164. IEEE, November 3-5, 1982.

[39] X. Yu, K. Chinomi, and K. et al. Privacy protecting visual processing for secure video surveillance. In *IEEE International Conference on Image Processing (ICIP 2008)*. IEEE Press, 2008.

Four
Privacy-Preserving User Clustering in a Social Network

This chapter has been published as “Privacy-Preserving User Clustering in a Social Network”, by Z. Erkin, T. Veugen, T. Toft and R. L. Lagendijk in the Proceedings of 1st *IEEE Workshop on Information Forensics and Security*, 2009.

Abstract

In a ubiquitously connected world, social networks are playing an important role on the Internet by allowing users to find groups of people with similar interests. The data needed to construct such networks may be considered sensitive personal information by the users, which raises privacy concerns. The problem of building social networks while user privacy is protected is hence crucial for further development of such networks. K-means clustering is widely used for clustering users in a social network. In this paper, we provide an efficient privacy-preserving variant of K-means clustering. The scenario we consider involves a server and multiple users where users need to be grouped into K clusters. In our protocol the server is not allowed to learn the individual user data and users are not allowed to learn the cluster centers. The experiments on the MovieLens dataset show that deployment of the system for real use is reasonable as its efficiency even on conventional hardware is promising.

4.1 Introduction

Internet applications in which people are grouped based on personal preferences have become very popular. By grouping users, these applications provide personalized services as well as building social networks where people can find the opportunity to communicate with others who share similar interests. There are a vast amount of social networks now available for dating, traveling, reading, cultural activities and many more [2]. Most users tend to give privacy sensitive data to benefit from such applications. As in the case of dating sites, the users provide to the system their personality details along with their preferences for a candidate while in traveling networks, the users announce a list of dates and locations for their planned travels.

The very success of applications based on finding similar people depends on the accuracy of grouping users which is directly proportional to the amount of collected user data. Since the content of the data is mostly privacy sensitive, the protection of the data is a raising concern among users [10]. Many rely on the trustworthiness of the service provider that possesses all the data. Several incidents have shown that this assumption is not completely true [1]. Even if the service provider protects its database against a common security problem of identity theft, there is no guarantee to prevent information being passed on without consent. A possible solution to protect the privacy sensitive data is having a trusted third party that is fully trusted by both the user and the service provider that keeps the data and runs the algorithm instead of the service provider. Unfortunately, having a third party that is fully trusted and willing to do all bulky computations is not realistic. A genuine solution is deploying cryptographic protocols to protect the privacy sensitive data of the users. Assuming that the server and the users are semi honest, meaning that they follow the protocol steps but are curious to extract more information than they allowed to have by storing all previous messages, it is possible to have a secure system where no information is revealed except the result of the algorithm run. With such a design, identity theft and abuse of user data by the service provider will be unlikely without having the secret key that is used to secure the user preferences.

A closer look at the problem of grouping users in a social network leads us to a well-known problem of clustering data. A user can be attached to a group of users if the user shares a common *taste* as of the users in that group. In a social network, the preference of a user is represented by a vector in the feature space. Thus, finding similar users with the same taste is basically a problem of clustering these feature vectors. The goal of the secure system is, then, grouping users with the same taste while protecting their privacy by hiding their preference data or feature vector. At the same time, the server should protect sensitive information about the algorithm like cluster locations. A malicious user can place himself into a desired cluster if this information is known. At the end of the secure clustering protocol, a user should only obtain the label information which is in fact a pointer to the cluster he is in, and the server should not get any information on the feature vector of the users.

As a method of clustering data, the K-means algorithm is widely used because of its simplicity and ability to converge extremely quickly in practice. Hence, in [3, 12, 11, 14] the authors addressed cryptographic techniques for the privacy-preserving clustering protocols based on K-means algorithm. In these works, the authors apply secure multiparty computation techniques [9], which makes any two-party privacy-preserving data mining problem solvable mostly by using Yao's secure circuit evaluation method [15]. Even though Yao's method can be used to implement any function in a privacy preserving manner, heavy computation costs in such circuits make these solutions feasible only for small circuit sizes which is a difficult requirement in many application scenarios. In [3, 12, 11], the authors attempt to solve the clustering problem in a two-party setting which is suitable to deploy techniques based on secret sharing. [12] suffers from a problem during the clustering algorithm where a division operation is misinterpreted as multiplication by the inverse which is not correct. On the other hand, [14] has a multi-user setting but requires three non-colluding entities for the clustering algorithm and the authors overcome the problem of updating centroids by allowing users to perform the division algorithm locally. In order to do that, the users possess the intermediate centroid assignments, meaning more information leakage. Therefore, these proposals are either not suitable for the problem of clustering users in a social network as they have different setting or not secure and efficient enough to deploy in practice.

In this paper, we provide a solution based on secure multi-party computation techniques in a semi-honest environment. Within this setting, our proposal groups people in a social network while protecting their privacy-sensitive data against the server and other users by means of encryption. A user gets a cluster identifier at the end of protocol but nothing more while the server obtains neither the identity of the users nor the content of user data. Our proposal provides a solution that is computationally efficient and scalable to a real life scenario of a centralized social network. We also show that communication cost of our protocol reaches the same performance of the most similar work in the field but achieving more privacy. The overall protocol was also implemented and tested exhaustively on the MovieLens dataset. Experimental results show that the algorithm proposed in this paper is both reliable and efficient for practical use.

4.2 Privacy-Preserving Clustering

Data clustering is a common technique for statistical data analysis where data is partitioned into smaller subgroups with its members sharing a common property [8]. Particularly, each user is represented as a point in an R dimensional space and is clustered according to minimal Euclidean distance. As a very common clustering technique, K-means assigns each user $P_i = (p_{i,1}, \ldots, p_{i,R})$ to the closest cluster among K clusters $C = \{C_1, \ldots, C_K\}$ where $C_j = (c_{j,1}, \ldots, c_{j,R})$. The algorithm starts with choosing the constant value K which is the number of clusters in the feature space. Each cluster is represented by its center (also named centroid) which is initially a random point in the space. In every iteration, the distance $D_{i,j}$ between i^{th} user P_i and cluster center C_j for $j = 1$ to K are calculated and the user is assigned to the cluster with the minimal distance. Once every user is assigned to a cluster, centroid locations are recalculated by taking the arithmetic mean of the user locations within each cluster. These two steps are repeated until either a certain number of iterations is reached or centroid locations are more or less fixed.

In the privacy-preserving version of the K-means clustering algorithm (Algorithm 3), each step is implemented in the encrypted domain. To realize this system, the server is assumed to have key pairs for himself of the Paillier [13] and Damgård, Geisler and Krøigaard (DGK) [5, 6] cryptosystems. These cryptosystems are chosen as they possess a property called *additive homomorphism* that allows us to process data in the encrypted domain such that the product of two encrypted values $[a]$ and $[b]$, corresponds to a new encrypted message whose decryption yields the sum of a and b as $[a] \cdot [b] = [a + b]$.

As a consequence of this additive homomorphism any ciphertext $[a]$ can be raised to the power b to obtain the encryption $[a]^b = [ab]$. In addition to the homomorphism property, Paillier and DGK cryptosystems are semantically secure implying that each encryption has a random element that results in different ciphertexts for the same plaintext. Throughout this paper we denote the Paillier encryption of a message m by $[m]$ and DGK encryption by $[\![m]\!]$. We omit the keys in the notation as all encryptions use the public key of the server. We also assume that the keys are generated and certified by a third trusted party (a certification authority) prior to starting of the protocol, and the public keys of the server are available to all users in the system. In the following sections, we give the details for each step of the Algorithm 3.

4.2.1 Computing Encrypted Distances

Assigning a user to the closest cluster requires Euclidean distance computations between a user P_i and centroid C_j in an R dimensional space as given in Equation 4.1. Regarding that the distance computations are only used for determining the minimum distance, taking the square root can be omitted.

Algorithm 3 The Privacy-preserving K-means clustering algorithm.

```
Require: The server sets parameter $K$ and selects $K$ random points as the initial
    centroids.
Ensure: A cluster pointer to the user.
1: The user computes encrypted distances to the $K$ current centroids.
2: The server and the user run an interactive protocol to find the minimum distance
   of $K$ encrypted distances to each centroid.
3: The server and all users jointly update the centroid locations.
4: Repeat step (1), (2) and (3) until the server finds that one of the termination con-
   ditions is reached.
5: The server and the user run a final protocol to reveal the cluster label to the user.
```

$$\begin{aligned} D_{i,j}^2 &= ||\mathbf{P}_i - \mathbf{C}_j||^2 = \sum_{n=1}^{R} (p_{i,n} - c_{j,n})^2 \\ &= \sum_{n=1}^{R} p_{i,n}^2 + \sum_{n=1}^{R} (-2p_{i,n}c_{j,n}) + \sum_{n=1}^{R} c_{j,n}^2. \end{aligned} \tag{4.1}$$

The implementation of the distance computation in the encrypted domain has two steps. First, the server encrypts (-2) times its centroid locations with his public Paillier key to obtain $[-2c_{j,n}]$ for all j and n, and publishes them. Then, the user calculates the encrypted Euclidean distance to each centroid as follows: the user computes the sum in first term in Equation 4.1 and encrypts it. In order to compute the encrypted second term, the additive homomorphism property of the Paillier cryptosystem is used. The user simply needs to raise each encrypted centroid value $[-2c_{j,n}]$ to the power of $p_{i,n}$ being the user's location in the n^{th} dimension. These values are then multiplied. Note that by receiving $[-2C]$ instead of $[C]$, the user spends less time for the costly exponentiation as he only uses $p_{i,n}$ as the power rather than $-2(p_{i,n})$. The calculation of the last term requires encryptions of the squares of the centroids. As all needed values are known to the server, it simply supplies these encryptions, $[\sum_{n=1}^{R} c_{1,n}^2], \ldots, [\sum_{n=1}^{R} c_{K,n}^2]$, along with the encryptions of $[-2C]$. Finally, the user multiplies these values to obtain the encrypted distance $[D_{i,j}^2]$ as given in Equation 4.2.

$$[D_{i,j}^2] = [\sum_{n=1}^{R} p_{i,n}^2] \cdot \prod_{n=1}^{R} [-2c_{j,n}]^{p_{i,n}} \cdot [\sum_{n=1}^{R} c_{j,n}^2]. \tag{4.2}$$

At the end of this step, each user possesses K encrypted distances from his location P_i to the current K centroids.

4.2.2 Preparing user data

After having obtained the encrypted distances to each centroid $[D_{i,1}], \ldots, [D_{i,K}]$, the i^{th} user needs to find the minimum of these K encrypted values. This requires a cryptographic protocol for comparing two encrypted values. Using the comparison protocol as shown in Section 4.3, the user obtains an encrypted vector $[\Gamma_i] = ([\gamma_{i,1}], \ldots, [\gamma_{i,K}])$ where $\gamma_{i,j}$ is 1 if and only if $D_{i,j}$ is the minimum distance (so user i is in cluster j), and 0 otherwise. Then, the user generates an encrypted matrix $[Z_i]$ to be used in updating the cluster centroids. This is simply the multiplication of vector Γ_i^T and user point P_i in the encrypted domain as shown in Equation 4.3.

$$[Z_i] = [\Gamma_i^T P_i] = \begin{vmatrix} [\gamma_{i,1}]^{p_{i,1}} & \ldots & [\gamma_{i,1}]^{p_{i,R}} \\ [\gamma_{i,2}]^{p_{i,1}} & \ldots & [\gamma_{i,2}]^{p_{i,R}} \\ \vdots & \vdots & \vdots \\ [\gamma_{i,K}]^{p_{i,1}} & \ldots & [\gamma_{i,K}]^{p_{i,R}} \end{vmatrix}, \tag{4.3}$$

where the j^{th} row of Z_i equals P_i when user i is in cluster j, and 0 otherwise.

4.2.3 Updating Centroids

Once all users complete their calculation on forming their encrypted vector $[\Gamma_i]$ and encrypted matrix $[Z_i]$, they jointly start a protocol for updating the centroids. For this step, the server creates a user chain as illustrated in Fig.4.1. We will explain the procedure for matrices Z_i, the accumulation of vectors Γ_i will be similar, and even simpler because the elements of Γ_i only take single bits.

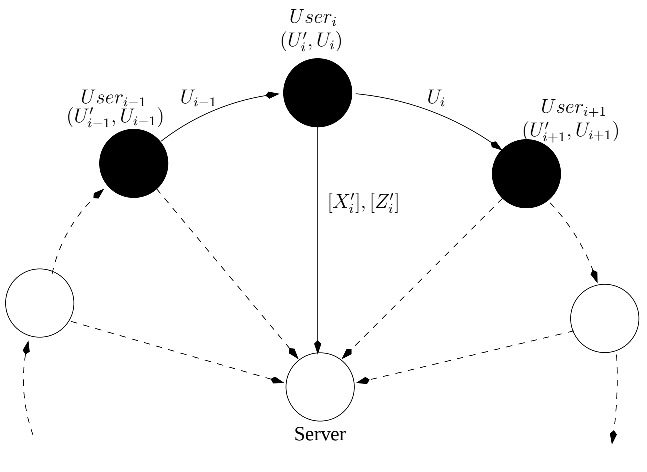


Figure 4.1: User chain created to update the cluster centroids.

Each user generates a random number r for each value in the matrix Z_i, to be used as blinding factors so the server will not learn the value of individual matrices. Actually for each user i, U_i is a K by R matrix of random values $(U_i)_{j,n}$. The matrix U_i is sent to the left neighbour of the user chain. Each user computes $(U_i)_{j,n} - (U_{i-1})_{j,n}$ as a blinding value for $(Z_i)_{j,n}$. Since the server will compute the sum

$Z^{sum}_{j,n} = \sum_{i=1}^{M}(Z_i)_{j,n}$, these random values will eventually cancel out. Note that the first user computes $(U_1)_{j,n} - (U_M)_{j,n}$, M being the number of users.

Suppose that user data $p_{i,n}$ is at most k bits. Then for each centroid j and user data index n, the sum $Z^{sum}_{j,n}$ can maximally take $k' = k + \lceil \log M \rceil$ bits. In order to sufficiently blind (a subchain of) the matrix elements, the random numbers should also be of size k'. In order to keep the blinding factors uniformly distributed, each user should compute $U_i - U_{i-1}$ modulo $2^{k'}$, before adding these to Z_i. Since the matrix Z_i is encrypted, the user cannot compute $Z_i + (U_i - U_{i-1})$ modulo $2^{k'}$. Therefore, an extra random number r' (or actually an extra matrix U' of random numbers) is needed to mask a possible overflow modulo $2^{k'}$. This extra random number should be κ bits where κ is a security parameter, to sufficiently mask the overflow to the server. Equation 4.4 shows the complete value $(Z'_i)_{j,n}$ that is sent to the server in encrypted form.

$$\begin{aligned}[(Z'_i)_{j,n}] &= [(Z_i)_{j,n}] \cdot [2^{k'}(U'_i)_{j,n} + \\ &\quad ((U_i)_{j,n} - (U_{i-1})_{j,n} \mod 2^{k'})]. \end{aligned} \tag{4.4}$$

The server will compute the matrix $[Z'^{sum}]$ by adding all matrix elements $[(Z'_i)_{j,n}]$ over all users i. Although the server could first decrypt the matrix elements, it would be more efficient to use the homomorphic property of Pallier:

$$\left[Z'^{sum}_{j,n}\right] = \left[\sum_{i=1}^{M}(Z'_i)_{j,n}\right] = \prod_{i=1}^{M}[(Z'_i)_{j,n}].$$

The server can simply compute the actual sum Z^{sum} by decrypting $[Z'^{sum}]$ and computing Z'^{sum} modulo $2^{k'}$ as shown in Equation 4.5. This is the sum of all user points per cluster.

$$\begin{aligned} Z'^{sum}_{j,n} &= \sum_{i=1}^{M}(Z_i)_{j,n} + 2^{k'}(U'_i)_{j,n} + ((U_i)_{j,n} - (U_{i-1})_{j,n}) \\ &= \sum_{i=1}^{M}(Z_i)_{j,n} + \sum_{i=1}^{M}2^{k'}(U'_i)_{j,n} \\ &= Z^{sum}_{j,n} + 2^{k'}\sum_{i=1}^{M}(U'_i)_{j,n} = Z^{sum}_{j,n} \mod 2^{k'}. \end{aligned} \tag{4.5}$$

A similar procedure is followed for the server to obtain X^{sum}, which is the number of users per cluster. The sum simply counts the number of ones, i.e. the number of users assigned to each cluster The server can then update the centroids by computing $c_{j,n} = Z^{sum}_{j,n}/X^{sum}_{j}$ and rounding the result to the nearest integer.

4.2.4 Termination Control and Getting User Labels

At the end of each iteration the server checks whether the predetermined termination condition is reached. Since centroids locations and the number of iterations are known to the server, this control is considered to be costless. Once the termination condition is reached, the label information of the user which is the index of the non-zero element in the encrypted vector $[\Gamma_i]$ should be revealed to the user. For this purpose, each user performs the following operation to obtain his cluster label information:

$$[\text{Id}] = \left[\sum_{j=1}^{K} (\gamma_{i,j} \times j) \right] = \prod_{j=1}^{K} [\gamma_{i,j}]^j, \tag{4.6}$$

where Id represents the cluster number that the user belongs to. Next, the user additively blinds this encrypted value with an uniformly random element r of size $\log(K)+\kappa$ to get $[\text{Id}+r]$ and re-randomize it before sending it to the server to be decrypted. The user can easily obtain his corresponding cluster label by subtracting r from the decrypted value sent by the server. This step completes the privacy-preserving K-means clustering algorithm.

4.3 Comparison Protocol

The most important building block in our protocol for privacy preserving K-means clustering is a cryptographic protocol that compares two encrypted ℓ bit values $[a]$ and $[b]$ and returns the minimum of these two values encrypted along with the result of the comparison $[\lambda]$ where λ is 1 if $a \geq b$ and 0 otherwise. Instead of using Yao's garbled circuit approach [15], which is often used and generally computationally expensive, in [7] a specialized fine-tuned protocol for this task is developed.

Having the comparison protocol in [7] at the core of our protocol, we build a binary tree for the values to be compared (see Section 4.2.2) as illustrated in Figure 4.2. We assume that K is a power of two. If this is not the case, dummy values can be added to the list of values to be compared. For K values, $K-1$ comparison results are stored by the user. When the comparisons are complete, each $[\lambda_{i,j}]$ is converted to DGK cryptosystem as this minor change improves the efficiency of the consequent computations considerably compared to using Paillier cryptosystem. For this conversion, the user computes a number $[\Lambda_i]$ composed of $[\lambda_{i,j}]$ as follows:

$$[\Lambda_i] = [\sum_{j=1}^{K-1} \lambda'_{i,j} 2^j] = \prod_{j=1}^{K-1} [\lambda'_{i,j}]^{2^j}, \tag{4.7}$$

where $[\lambda'_{i,j}]$ is either $[\lambda_{i,j}]$ or $[1-\lambda_{i,j}]$ with probability 0.5 to hide the comparison results from the server. Upon receiving the value $[\Lambda_i]$, the server decrypts it and encrypts every bit value $\lambda'_{i,j}$ using DGK cryptosystem to obtain $[\![\lambda'_{i,j}]\!]$ and sends them back to the user. The user then reverses the hiding procedure by computing either $[\![\lambda'_{i,j}]\!]$ or $[\![1-\lambda'_{i,j}]\!]$ to obtains the correct values. After this conversion between cryptosystems, the user assigns the values $[\![\lambda_{i,j}]\!]$ and $[\![1-\lambda_{i,j}]\!]$ to the left and right branches of each node of the tree respectively (Figure 4.2). Next, the user traverses the tree from root to top to reach each leaf while adding up the branch values which corresponds to the multiplication of the encrypted values. At the end of this procedure, we obtain an encrypted value $[\![\zeta_{i,j}]\!]$ for each leaf, thus $D_{i,j}$. Only for the minimum $D_{i,j}$ the value $\zeta_{i,j}$ is zero since all the branch values in the path should be zero. For the others, $\zeta_{i,j}$ is a non-zero value.

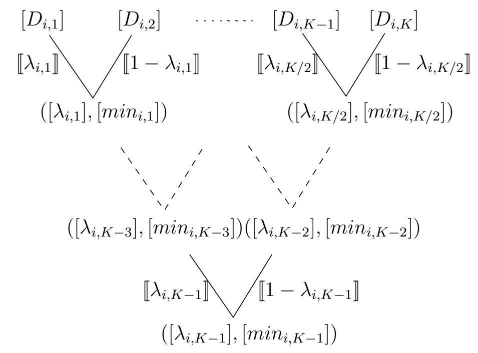

$[D_{i,1}]$
$[D_{i,2}]$
$[D_{i,K-1}]$
$[D_{i,K}]$
$[\![\lambda_{i,1}]\!]$
$[\![1-\lambda_{i,1}]\!]$
$[\![\lambda_{i,K/2}]\!]$
$[\![1-\lambda_{i,K/2}]\!]$
$([\lambda_{i,1}],[min_{i,1}])$
$([\lambda_{i,K/2}],[min_{i,K/2}])$
$([\lambda_{i,K-3}],[min_{i,K-3}])$
$([\lambda_{i,K-2}],[min_{i,K-2}])$
$[\![\lambda_{i,K-1}]\!]$
$[\![1-\lambda_{i,K-1}]\!]$
$([\lambda_{i,K-1}],[min_{i,K-1}])$

Figure 4.2: Binary tree used to form user vector X_i.

This set of values is then re-randomized, and permuted with a uniformly random permutation π before sending them to the server. The server decrypts the values and creates a new vector that only has a one value at the same position of zero in the received vector and zeros everywhere else. This vector is then encrypted item-wise with Paillier public key and sent back to the user. After repermutation, it is used as the X_i vector described in Section 4.2.2.

4.4 Security (Sketch)

We show that our clustering algorithm fulfills the privacy claims: 1) The server learns the number of people per cluster and the accumulated personal data per cluster, but not the personal data of separate users. 2) The users learn the number of iterations of the clustering algorithm and the index number of his own cluster, but not the cluster centroids. We give a short sketch that justifies our privacy claims in a semi-honest attack model.

Ad 1. The server performs the comparison protocol with each user many times in order to determine the cluster centroid that is closest to each user. However, this comparison protocol is designed in such a way that the server will learn no information about the user data [7] or the actual minimum (index). After the comparison protocol is executed K-1 times, the K-1 comparison results have to be combined in order to find the minimum distance to the current K cluster centroids. In order to switch from Pallier to DGK, the server is given a blinded combination of all comparison results (see Equation 4.7) that keeps the values of all K-1 comparison results hidden to the server.

The DGK encrypted comparison results are then used to compute K DGK encrypted values, one for each distance, in order to obtain the vector X_i. When sent to the server, these values are permuted but not blinded. This is sufficient, as the server already knows the values it will receive, and their order is kept secret by the permutation. This claim is easily proven by induction in the height of the tree. For $K = 2$, the server will receive encryptions of 0 and 1. For $K = 2^{\alpha+1}$, the tree contains two subtrees of equal size with values known to the server. The $\zeta_{i,j}$'s of the leaves of one of those subtrees will be increased by one, but the server is still able to predict the values seen.

When updating the centroids, the server obtains data from each user. Just like in the Dining Cryptographers problem [4], each user shares his random number with his neighbour thereby masking the individual data from both users. The server will learn nothing by looking at a sum for some sub-chain of users, he will only obtain the full accumulated data because only then all random numbers cancel out. Intuitively, each user masks his input along with the inverse of the mask of his neighbour. When eventually computing the cluster index for each user, each user will blind the index by a random number to avoid information leakage to the server.

Of course, when the number of users is small, the accumulated data will reveal some information about the personal data, but still the server will not know which personal data came from which user.

Ad 2. Each user learns the values $[C]$ and $[C^2]$ of the (intermediate) cluster centroids, and from these encrypted distances $[D_{i,j}]$ to each cluster centroid can be computed. But since Paillier encryption is semantically secure, this will not leak information about C, C^2, or $D_{i,j}$. The comparison protocol, whose security is proven in [7], results in an encrypted comparison bit and the encrypted minimum. Since all further information that is obtained in the computation of the encrypted user vector Γ_i is also encrypted, this will also not leak information. While updating the centroids, users obtain random numbers from neighbours which are used to blind the user vectors from the server. Since no personal data is exchanged here, again no information is leaked.

Since each user knows the number of iterations, some information is leaked about the cluster centroids, especially when the total number of users is small. But in practice this amount can probably be neglected [12].

4.5 Experiments

The privacy-preserving K-means clustering algorithm presented in the paper has been implemented in order to determine its performance and reliability. This has been done in C++ using the GNU GMP library version 4.2.1. The tests were performed on a computer with Intel Xeon 2.33 GHz processor and 32GB of Ram running SuSE 10.3 operating system. Both server and user, modeled as separate classes, run on the same machine, thus network latency was not considered in the test results.

The MovieLens dataset (`www.grouplens.org`) was used for our experiments. This contains 100,000 integer ratings in the range of $[0, 5]$ for 1682 movies by 943 users. As the sparseness of the dataset is great (94%), a subset containing the movies rated by most users was considered. We filled the null entries of this subset with the user mean vote rounded to the nearest integer value for the corresponding row. The number of movies in this subset, represented by R, also determines the parameter ℓ which is the bit length of the values to be compared. The parameter ℓ should be big enough to hold the largest possible value which is the Euclidean distance squared between two user rating vectors in our case. Since our aim is to show the equivalent accuracy between the plain and privacy-preserved K-means clustering algorithm, we set the number of clusters K to a single value. The parameters used for the experiments are given in Table 4.1.

Table 4.1: Parameters.

Parameter	Value
R	12
K	10
ℓ	9 bits
κ	112 bits
Paillier Encryption	2048 bits
DGK Encryption	1024 bits

4.5.1 Reliability

The privacy-preserving K-means clustering protocol is designed for an accuracy equivalent to the plain clustering algorithm. The only possible degradation is due to the integer arithmetic used for updating centroids. As the cryptosystem used in our protocol accepts only integer values, the location of the centroids in the space should also be represented by integers. This can be achieved by scaling and rounding the values, however, in our application scenario we experienced that using scaling does not introduce noticeable improvement that can effect the outcome of the clustering protocol. Therefore, we only round the values to the nearest integer.

4.5.2 Round Complexity

The distance computation and the updating of the centroids both require one round, while each comparison require four. As the minimum of K values can be found in $\mathcal{O}(\log(K))$ rounds by using binary tree approach as illustrated in Figure 4.2, the overall round complexity of one iteration of the privacy-preserving K-means clustering protocol is $2 + 4\log(K)$. In addition, one extra round is required at the end of the clustering protocol to send the final labels to the users. The round complexity of our work outperforms the comparable work of Vaidya and Clifton [14] which has a round complexity of $\mathcal{O}(M + K)$ in their basic algorithm and $\mathcal{O}(M)$ in optimized version.

4.5.3 Communication Complexity

Communication complexity is mainly determined by the amount of encrypted messages exchanged between the server and the users. Therefore, it is related to the size of the Paillier and DGK encryptions. The communication complexity is $\mathcal{O}(K(R+\ell))$. Considering that $\ell = \log_2(R \times c)$ where c is the possible maximium distance-squared between two rating vectors of size R, the overall communication cost can be written as $\mathcal{O}(KR)$. The amount of data sent by the server and the user is 48 kB each for one iteration using the parameters given in Table 4.1.

As [3, 12, 11] have proposals for a two-party setting based on secret sharing, we can compare our result only to [14] which has the same communication cost of $\mathcal{O}(KR)$ bits. In addition to achieveing the same level of communication cost, in our proposal, we keep intermediate centroid locations hidden from the users and have no need for particular non-colluding entities, meaning better privacy.

4.5.4 Computational Complexity

The computational complexity of our privacy-preserving K-means clustering protocol is mainly dependend on the Paillier and DGK cyptosystems. In one iteration of the clustering, the total computational complexity is $\mathcal{O}(KR)$ Paillier encryptions, exponentiations and multiplications and $\mathcal{O}(K\ell)$ DGK encryptions and $\mathcal{O}(K\ell^2)$ DGK multiplications for the user. The server needs to compute $\mathcal{O}(K+R)$ Paillier encryptions, $\mathcal{O}(KR)$ Paillier decryptions and $\mathcal{O}(MKR)$ Paillier multiplications. In addition to that, the server also computes $\mathcal{O}(K\ell)$ DGK encryptions and decryptions.

The running time of our implementation is given in Table 4.2 with a different number of users for 10 iterations. As given in the complexity analysis, for constant K and R the running time is linear in the number of users M in the system and it only takes roughly 1 hour to cluster 943 users.

Table 4.2: Computational Complexity (in minutes).

M	Time	Time with pre-computation
100	17.9	6.9
250	44.1	17.2
500	88.0	33.8
750	134.5	51.6
943	166.1	64.5

The second column of Table 4.2 reflects the running time of the whole protocol without any optimization. The random values used for encryption and blinding and the exponentiations r^n, h^r that are used in Paillier and DGK encryptions respectively can be generated in idle processor time or prior to the start of the protocol. The running time of the protocol under these optimizations is given in the third column of Table 4.2. These values will be much smaller in a real system where there are at least $M+1$ processors and many operations can be realized asynchronously.

4.6 Variations

The privacy-preserving K-means clustering algorithm introduced in this paper can be improved further in two main directions. First, the efficiency of the protocol can be enhanced by employing simple modifications. The amount of data sent among users can be reduced significantly, resulting less communication cost. In each iteration, the users receives several random values from their right neighbour to be used in blinding procedure for updating the centroids. Instead of random values, the users can exchange keys for the pseudo-random generator (PNG) in the first place so that they can generate the random values themselves. It is also important to note that depending on the type of user data, a considerable amount of computations can be avoided. Since random values depends on the bit-length of user data k, for small values of k, small random values are needed. This introduces a considerable gain in computations.

Second, the privacy of the users can be protected more by benefiting from the structure of social networks. If the server would collude with two users, the user chain is divided and the server effectively splits the user set in two. However, the blinding for the centroid update procedure does not require a random user chain, merely a connected graph, where pairs of users blind/unblind. Instead of having a random user chain, one idea could be to use a “friend-list” from the site. Not only does this make it more difficult for the server to split the group in two, people may have more trust in those they know rather than some other arbitrary users of the system.

4.7 Conclusion

In this paper we present an efficient, privacy-preserving K-means clustering algorithm in a social network setting. In particular, we propose a protocol in which privacy sensitive data of the users is kept hidden from the server and the cluster locations in the user space are kept secret from the users. Our protocol mainly uses secure multiparty computation techniques, but instead of using generic solutions, it benefits from fine-tuned cryptographic protocols developed for higher efficiency. Our proposal achieves more privacy by hiding all sensitive user data from the server and the centroid locations from the users with the same level of communication cost of [14] which has a comparable multi-user setting. The implementation of the privacy-preserving K-means clustering algorithm with MovieLens dataset shows that the accuracy of the system is as reliable as the reference implementation in clear and the running time of the protocol is promising even on modest hardware platforms. The numbers we have obtained from the experiments show that the protocol presented in this paper is efficient enough to be deployed in practice.

References

[1] Monster attack steals user data. BBC News, August 2007, http://news.bbc.co.uk/2/hi/technology/6956349.stm.

[2] Social networking: A quantitative and qualitative research report into attitudes, behaviours and use. Ofcom, April 2008, http://www.ofcom.org.uk/advice/media_literacy/medlitpub/medlitpubrss/socialnetworking/report.pdf.

[3] P. Bunn and R. Ostrovsky. Secure two-party k-means clustering. In *Proceedings of the 14th ACM conference on computer and communications security*, pages 486–497. ACM New York, NY, USA, 2007.

[4] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1(1):65–75, 1988.

[5] I. Damgård, M. Geisler, and M. Krøigaard. Efficient and Secure Comparison for On-Line Auctions. In *Australasian Conference on Information Security and Privacy — ACSIP 2007*, volume 4586 of *LNCS*, pages 416–430. Springer, July 2-4, 2007.

[6] I. Damgård, M. Geisler, and M. Krøigaard. A correction to "Efficient and secure comparison for on-line auctions". Cryptology ePrint Archive, Report 2008/321, 2008. http://eprint.iacr.org/.

[7] Z. Erkin, M. Franz, S. Katzenbeisser, R. L. Lagendijk, J. G. Merchan, and T. Toft. Privacy-preserving face recognition. In *Proceedings of the Privacy Enhancing Technologies Symposium*, Seattle, USA, 2009(accepted).

[8] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, 1990.

[9] O. Goldreich. *Foundations of Cryptography. Basic Applications*, volume 2. Cambridge University Press, first edition, May 2004.

[10] S. Harris. Security issues of social network sites. InformIT, February 2009, http://www.informit.com/blogs/blog.aspx?uk=Security-Issues-of-Social-Network-Sites.

[11] G. Jagannathan, K. Pillaipakkamnatt, and R. Wright. A New Privacy-Preserving Distributed k-Clustering Algorithm. In *Proceedings of the Sixth SIAM International Conference on Data Mining*, 2006.

[12] G. Jagannathan and R. N. Wright. Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pages 593–599, New York, NY, USA, 2005. ACM.

[13] P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In J. Stern, editor, *Advances in Cryptology*, volume 1592 of *LNCS*, pages 223–238. Springer, May 2-6, 1999.

[14] J. Vaidya and C. Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *KDD ’03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 206–215, New York, NY, USA, 2003. ACM.

[15] A. C.-C. Yao. How to Generate and Exchange Secrets (Extended Abstract). In *Annual Symposium on Foundations of Computer Science — FOCS ’85*, pages 162–167. IEEE, October 27-29, 1986.

Five

Privacy-Preserving Recommender System

This chapter is based on the paper “Privacy-Preserving Centralized Recommender System”, Z. Erkin, T. Veugen, T. Toft and R. L. Lagendijk, the *IEEE Transactions on Information Forensics and Security*, (in preparation) 2010.

Abstract

Recommender systems have become an important research area as they enable personalized recommendations and services to users. These systems which explore user behavior and user ratings to improve the recommendation process are highly dependent on the amount of information collected from the users. This information is mostly privacy-sensitive and open to be abused by the service provider himself if not protected properly. In this paper, we propose a method based on cryptographic techniques to provide privacy to the users of recommender systems. To achieve this goal, the user data are kept encrypted on a server and a third actor called Privacy Service Provider (PSP) participates in a secure protocol to generate recommendations for the users. The proposed protocol does not leak information to the PSP and the server, providing privacy to the users. The common problem of data expansion due to the use of public key cryptosystems for encryption is handled by packing values throughout the protocol. This approach reduces both communication and computation costs significantly as shown in performance analysis.

5.1 Introduction

Recent statistics show that more than 48% people in Europe, 60% in Oceania and 74% in North America have Internet access by the end of year 2008 [1]. These numbers are still growing as people start connecting to the Internet not only by using PCs but also having portable gadgets like mobile phones and PDAs. A closer look at the usage of the Internet shows that many people benefit from numerous applications that enable them to communicate with other people, do online shopping, find locations and download digital content. The numbers for some social network sites can demonstrate the wide usage of such applications: Facebook 250 million, MySpace 260 million, LinkedIn 43 million and Adult FriendFinder 33 million registered users [2].

The business model of such applications aims at increasing the number of its users by providing high quality services. In order to achieve that, service providers offer personalization so that the users can customize the service according to their preferences. Depending on the previous behavior or likes-dislikes, the application adapts its algorithm to the user profile. As in the case of online shopping, the user can be suggested a list of items which is derived from the user's previous shopping list or the items that are bought by similar users. In the case of social network sites, user preference data can be used to find similar people or groups. Because of its impact on e-business and research challenges, recommender systems have become an important research area since mid-1990s both in the industry and academia. In spite of the diversity of techniques on generating recommendations [3], they all rely on the same basis: gathering more information about the users.

Depending on the application type, the required data can vary from name, age, birth date and location, traveling plans, plate number to more privacy-sensitive data like medical records. It is quite possible that the service provider may try to collect more information on the users or even if this is not the case, the stored data can be attacked by intruders. In either case, the consequences are severe for the privacy of the users. Therefore, there is no solid guarantee on the privacy of user data against neither the service provider nor the adversaries. Even though the law and the regulations in the direction of protecting user privacy limit the service providers in repurposing the data to their own benefits, without privacy-sensitive technologies these regulations do not provide full privacy.

Privacy problems in recommender systems have been raised in several works. In [4], Canny proposes a system where the private user data is encrypted and recommendations are generated by applying an iterative procedure based on conjugate gradient algorithm. The algorithm computes a characterization matrix of the users in a subspace and generates recommendations by calculating reprojections on it in the encrypted domain. Since the algorithm is iterative, it takes many rounds for convergence and in each round users need to participate in an expensive decryption procedure which is based on a threshold scheme where a significant portion of the users are assumed to participate and be honest. The output of each iteration which is the characterization matrix is available in clear. In [5], Canny proposes a method to protect the privacy of users based on a probabilistic factor analysis model by using a similar approach as in [4].

While Canny prefers to work with encrypted user data, Polat and Du suggest to protect the privacy of users by using randomization techniques [13, 14]. In their paper, they blind the users data with a known random distribution assuming that in aggregated data this randomization will cancel out and the data obtained will be a good estimation of the intended original data. The success of this method highly related to the number of users participating in the computation and this creates a trade-off between accuracy/correctness of the recommendations and number of users. The outcome of the algorithm is also available to the server which aggregates the data. In addition to this information leakage, the randomization techniques are believed to be highly insecure [16].

We propose to encrypt the user preference data by using a homomorphic cryptosystem. Once the data is encrypted, it is sent to the service provider that has a business interest for generating recommendations to the users. Since the data is encrypted, processing it requires using cryptographic protocols based on secure multiparty computation techniques which are mostly interactive. In order to limit the user's participation in such interactive protocols and thus to reduce the workload of the user, we propose a new actor in our privacy-preserving recommender system, namely *Privacy Service Provider* (PSP). The PSP is different from trusted third parties (TTP) in the sense that the PSP has a business interest in providing recommendations to the users. Therefore, the proposed system consists of two entities: 1) the server that stores the encrypted user data and 2) the PSP that participates in generating recommendations in a privacy-preserving manner in accordance with the server. Here, it is important to note that the PSP and the server have different business interest such that the PSP is interested in generating recommendations for the user whereas the server offers safe storage in vast amounts.

To illustrate this idea, consider the following example. A user rates several restaurants; travels to another city and wants to find a good restaurant. For this purpose, he asks for a recommendation from the PSP based on his rating stored by the server.

The outcome of the protocol is average ratings from a group of similar other people for some restaurants. Regardless of the application, in our protocol the content of user data, the intermediate values and the output of the algorithm are unknown to the PSP and the server. This provides better privacy compared to [4, 5]. In addition to that, our protocol is based on provably secure cryptographic primitives and does not depend on the number of users which is not the case in [13, 14].

Since our technique to protect the privacy of users involves semantically secure asymmetric cryptosystems; an expansion in data size is inevitable which increases the communication cost. To reduce the cost, we propose packing user data when it is possible. As a result, the amount of expensive operations on encrypted data reduces considerably. The dramatic decrease in both computation and communication costs makes our system particularly promising for real system deployment as shown in performance analysis.

5.2 Collaborative Filtering

A centralized system for generating recommendations is a common approach in e-commerce applications. To generate recommendations for a user, the server follows a two-step procedure. In the first step, the similar users in the system are searched. Each user in the system is represented by a preference vector which is usually formed of ratings for each item within a certain range. Finding similar users is based on computing similarity measures between users' preferences vectors. The similarity measure is a Pearson correlation as defined in (Eq. 5.1) for two users with preference vectors $V_A = (x_{A,0}, \ldots, v_{A,M-1})^T$ and $V_B = (v_{B,0}, \ldots, v_{B,M-1})^T$ respectively where M is the number of items and, $\overline{v}$ represents the average value of the vector x.

$$\text{sim}_{A,B} = \frac{\sum_{i=0}^{M-1}(v_{A,i} - \overline{v}_A)(v_{B,i} - \overline{v}_B)}{\sqrt{\sum_{i=0}^{M-1}(v_{A,i} - \overline{v}_A)^2 \sum_{i=0}^{M-1}(v_{B,i} - \overline{v}_B)^2}}. \tag{5.1}$$

Once the similarity measures between users are computed, the server proceeds with the second step. In this step, the server chooses the first L users with the highest similarity values and determines the *recommendation* by averaging their ratings for the requested item.

In e-commerce applications the number of items offered to users are usually in the order of hundreds of thousands. Apart from many smart ways of determining the likes and dislikes of users for the items such as click log analysis, we assume the users are asked to rate the items explicitly with integer values in the range of $[0, K]$. Regarding the number of items and user behavior for rating these, the data that the server can obtain is highly sparse, meaning that most of the items are not rated. Finding similar users in a sparse dataset can easily lead the server to generate inaccurate recommendations. To cope with this problem, one approach is introducing a small set of items that is rated by most users. Such a base set can be explicitly given to the users or implicitly chosen by the server from mostly rated items. Having a small set of items that is rated by most users, the server can compute similarities between users more confidently, resulting in more accurate recommendations. Therefore, we assume that the user preference vector V is split into two parts: the first part consists of R elements that are fully rated by most of the users and the second part contains $M - R$ partly rated items that the user would like to get recommendations on [3].

5.3 Preliminaries

We use encryption to protect user data against the recommender system, i.e. the server and other users. A special class of cryptosystems, namely homomorphic cryptosystems, allow us to process the data in its encrypted form. In this section we briefly describe the homomorphic cryptosystems and introduce two cryptographic protocols that we use in our privacy-preserving centralized recommender system. We use the semi-honest security model, which assumes that all players follow the protocol steps but are curious and thus keep all messages from previous and current steps to extract more information than they are allowed to have. Our protocol can be adapted to the active attacker model by using the ideas in [11] with an additional overhead.

5.3.1 Homomorphic Cryptosystems

We use two cryptosystems: Paillier [12] and Damgård, Geisler and Krøigaard (DGK) [7, 8]. We use the Paillier cryptosystem to encrypt the privacy-sensitive data whereas DGK is used in a cryptographic subprotocol that is particularly designed to compare encrypted values. This protocol makes computations on bit level and the DGK cryptosystem was chosen as it performs better than the Paillier cryptosystem in terms of encryption and decryption time due to its much smaller message space.

The Paillier and the DGK cryptosystems possess a property called *additive homomorphism* that allows us to process data in the encrypted domain: the product of two encrypted values $[a]$ and $[b]$ where $[\cdot]$ denotes the encryption function, corresponds to a new encrypted message whose decryption yields the sum of a and b as $[a]\cdot[b] = [a+b]$.

As a consequence of the additive homomorphism any ciphertext $[a]$ raised to the power b results in the encryption $[a]^b = [a \cdot b]$. In addition to the homomorphism property, the Paillier and the DGK cryptosystems are semantically secure implying that each encryption has a random element that results in different ciphertexts for the same plaintext. Throughout this paper we denote the Paillier encryption of a message m by $[m]$ and the DGK encryption by $[\![m]\!]$. We omit the keys in the notation as all encryptions use the public key of the PSP.

5.3.2 Secure Multiplication Protocol

The secure multiplication protocol in [6, 9] can be adapted to a two-party protocol in which one party, A, has two encrypted values $[a]$ and $[b]$, and the other party, B, has the decryption key. The protocol outputs the encrypted value $[a \cdot b]$ to the party A without B learning a or b. Assuming that the encryption scheme is additively homomorphic,

1. A generates two uniformly distributed random numbers r_1 and r_2, and subtracts these numbers from the encryptions $[a]$ and $[b]$ respectively: $[\tilde{a}] = [a] \cdot [-r_1] = [a - r_1]$, $[\tilde{b}] = [b] \cdot [-r_2] = [b - r_2]$. Afterwards, he sends $[\tilde{a}]$ and $[\tilde{b}]$ to B.

2. B decrypts $[\tilde{a}]$ and $[\tilde{b}]$, multiplies them and sends the encrypted product, $[\tilde{a} \cdot \tilde{b}]$ to A.

3. A removes the random values and obtains the encryption of the product of a and b as follows: $[a \cdot b] = [\tilde{a} \cdot \tilde{b}] \cdot [a]^{r_2} \cdot [b]^{r_1} \cdot [-r_1 \cdot r_2]$.

5.3.3 Secure Decryption Protocol

Similar to the secure multiplication protocol, a secure decryption protocol can be designed based on [6]. In this protocol, party A demands for the decryption of an encrypted value, $[a]$ without revealing it to the owner of the decryption key B. Using an additively homomorphic cryptosystem, this protocol can be summarized as follows:

1. A generates a uniformly random number r and blinds the encryption with this number: $[\tilde{a}] = [a] \cdot [r]$. Then, A sends $[\tilde{a}]$ to B.

2. B decrypts $[\tilde{a}]$ and sends it back to A.

3. A obtains the decryption of $[a]$ by subtracting r from $\tilde{a}$: $a = \tilde{a} - r$.

5.4 Privacy-Preserving Collaborative Filtering

We propose a cryptographic protocol based on secure multiparty computation (SMC) techniques to implement the two steps of the recommendation procedure introduced in Sect. 5.2 in a privacy-preserving manner. The privacy-sensitive data of users, i.e. preferences, is stored by the server in the encrypted form. During the recommendation generation, the intermediate outcomes of the protocol, namely similarity values among users, are privacy sensitive and must be kept secret both from the server and the PSP. The identity of the most similar users are, of course, unknown to all three players.

Our protocol starts with a request from a user for recommendations. Upon the request, the PSP and the server initiates a protocol to determine the similarity values between the user requesting recommendations and the others in the system. The L most similar users are chosen for the second step in which their ratings are accumulated under encryption. In the final step, the server sends the accumulated ratings and the number L to the user. The user obtains the desired recommendations after dividing the accumulated rating by L. This protocol is detailed in the following sections.

5.4.1 Step 1: Initialization

The preference data and similarity measures are protected by means of encryption. For this purpose, the PSP generates key pairs for the Paillier and the DGK cryptosystems and publishes the public keys with valid certificates. Given the Paillier public key, all users encrypts their preference data as follows. The first R ratings that are to be used for computing the similarities are processed and scaled to be used in similarity computation. Since the Pearson correlation given in (5.1) for user A and B can be also written as:

$$\mathrm{sim}_{A,B} = \sum_{i=0}^{R-1} \underbrace{\frac{(v_{A,i} - \overline{v}_A)}{\sqrt{\sum_{i=0}^{R-1}(v_{A,i} - \overline{v}_A)^2}}}_{C_1} \cdot \underbrace{\frac{(v_{B,i} - \overline{v}_B)}{\sqrt{\sum_{i=0}^{R-1}(v_{B,i} - \overline{v}_B)^2}}}_{C_2}, \tag{5.2}$$

the terms C_1 and C_2 can be easily computed by users A and B, respectively. Each user computes a vector from which the mean is subtracted and normalized. Since the elements of the vector are real numbers and cryptosystems are only defined on integer values, they are all scaled by a parameter f with enough precision and rounded to the nearest integer resulting in a new vector $V_i' = (v_{i,0}', \dots, v_{i,R-1}')^T$ whose elements are now k bit positive integers. The remaining elements of the vector V_i are processed to have a packed representation. We follow a similar construction to [15] to pack values in one encryption to decrease the number of encryptions to be transferred between the server and the PSP and the number of operations on the encrypted data. For simplicity, we assume that packing arbitrary number of values in one encryption is possible. We clarify this later in this section.

The vector elements of V_i' are packed in such a way that it can allow addition of L values of size k bits as follows:

$$\alpha_i = \sum_{j=0}^{M-R-1} v_{i,j+R} \cdot (2^{k+\log(L)})^j, \tag{5.3}$$

where i is the user index, k is the number of bits used to represent the ratings after scaling and L is the upper bound of the number of most similar users above a threshold δ in the system. All N users in the system encrypt these computed values with the Paillier public key of the PSP and send $([v_{i,0}], [v_{i,1}], \dots, [v_{i,R-1}], [\alpha_i])$ to the server. Note that the decryption key is only available to the PSP, meaning the server cannot decrypt and see the content of the encryptions.

Each user that enters the recommender system is required to upload his encrypted data to the server. After that, any user is able to request for recommendations by notifying the server.

5.4.2 Step 2: Finding Similar Users

Upon the request of recommendations from user A, the PSP and the server initiate a protocol to find the best L similar users to the user A. Similar users can be found by computing similarity measures between the user A and all other users in the system as described in Sect. 5.2. Note that after processing the first R elements of the user preference vector as described in the initialization step, the similarity computation based on Pearson correlation becomes an inner product of two vectors as given below.

$$[\mathrm{sim}_{A,B}] = [(V_A')^T \cdot V_B'] = [\sum_{j=0}^{R-1} v_{A,j}' \cdot v_{B,j}']. \tag{5.4}$$

Because only the encrypted forms are available to the server, the similarity computation requires running the secure multiplication protocol as described in Sect. 5.3. This protocol consists of several encryptions, decryptions, exponentiations and multiplications. Instead of running the secure multiplication protocol for each similarity computation between user A and user i, the server creates a packed representation σ_j^A of the all $[V_i']$'s to reduce the costs as follows (Fig. 5.1):

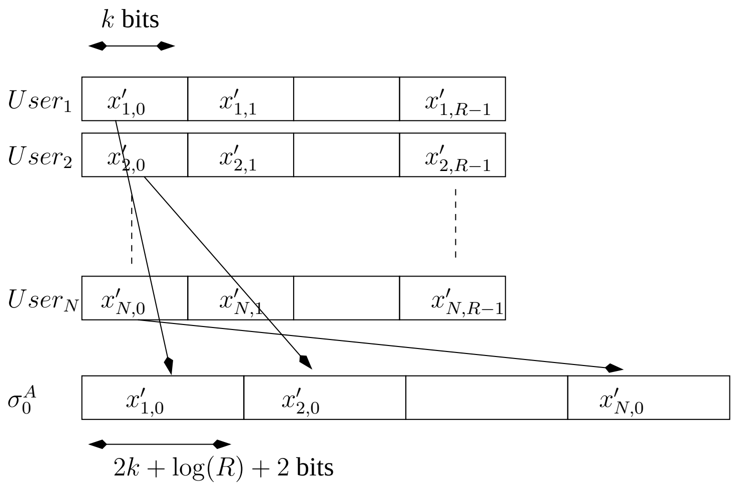


Figure 5.1: Illustration of packing $[V_i']$'s. The values with the same index are packed for all users in one encryption with enough space to allow further processing.

$$[\sigma_j^A] = [\sum_{i=0, i\neq A}^{N-1} 2 \cdot v'_{i,j} \cdot (2^{2k+\log(R)+2})^i] = \prod_{i=0, i\neq A}^{N-1} [v'_{i,j}]^{2\cdot(2^{2k+\log(R)+2})^i}, \quad (5.5)$$

for $j = 0$ to $R-1$ where $2k+\log(R)+2$ is the required number of bits for multiplying two k-bit numbers and adding R of them. The additional 2 bits are necessary for the procedure described in Sect. 5.5.

Once the server computes the values $([\sigma_i^A]$, he runs R secure multiplication protocols to obtain $([\sigma_0^A \cdot v'_{A,0}], \ldots, [\sigma_{R-1}^A \cdot v'_{A,R-1}])$. The server then computes the encrypted sum of these values $[\Sigma_A]$,

$$[\Sigma_A] = [\sum_{j=0}^{R-1} \sigma_j^A \cdot v'_{A,j}] = \prod_{j=0}^{R-1} [\sigma_j^A \cdot v'_{A,j}] = [\text{sim}_{A,0}|\text{sim}_{A,1}|\ldots|\text{sim}_{A,N-1}]. \quad (5.6)$$

The value $[\Sigma_A]$ is in fact the packing of N similarity values (Fig. 5.2). After having computed the similarity values between user A and every other user in a packed form, the server runs a protocol with the PSP to determine a set of users with high similarity. This cryptographic protocol, as detailed in Sect. 5.5, hides the content of the encrypted

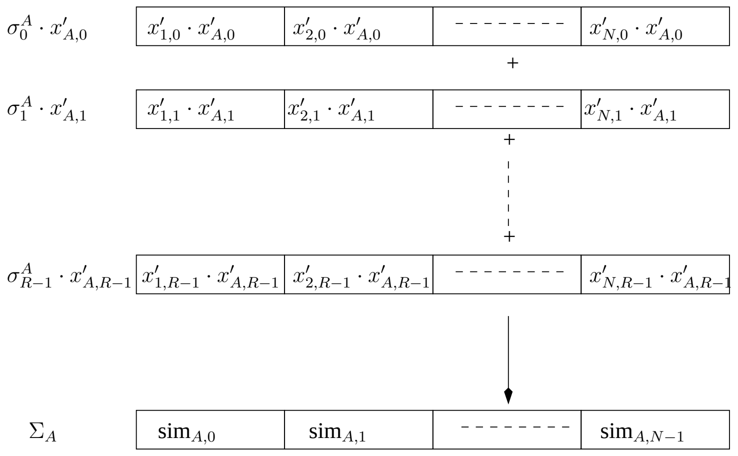

$\sigma_0^A \cdot x'_{A,0}$
$x'_{1,0} \cdot x'_{A,0}$
$x'_{2,0} \cdot x'_{A,0}$
$x'_{N,0} \cdot x'_{A,0}$
$+$
$\sigma_1^A \cdot x'_{A,1}$
$x'_{1,1} \cdot x'_{A,1}$
$x'_{2,1} \cdot x'_{A,1}$
$x'_{N,1} \cdot x'_{A,1}$
$+$
$+$
$\sigma_{R-1}^A \cdot x'_{A,R-1}$
$x'_{1,R-1} \cdot x'_{A,R-1}$
$x'_{2,R-1} \cdot x'_{A,R-1}$
$x'_{N,R-1} \cdot x'_{A,R-1}$
Σ_A
sim$_{A,0}$
sim$_{A,1}$
sim$_{A,N-1}$

Figure 5.2: Illustration of packing σ_j^A's to obtain Σ_A in clear.

values from both the server and the PSP, and outputs a vector of encrypted values $[\Gamma_A] = ([\gamma_{A,0}], [\gamma_{A,1}] \ldots, [\gamma_{A,N-1}])$ where $\gamma_{A,i}$ is 1 if and only if the sim$_{A,i}$ exceeds a value δ and 0 otherwise.

5.4.3 Step 3: Generating Recommendations

Once the encrypted vector $[\Gamma_A]$ is obtained, the server runs the secure multiplication protocol with the PSP to multiply each α_i with the corresponding $\gamma_{A,i}$ value. These multiplications yield the values $[\Phi_i] = [\alpha_i \cdot \gamma_{A,i}]$ for $i = 0$ to $N - 1$. Note that Φ_i will be equal to the encryption of α_i when $\gamma_{A,i} = 1$ and contains an encrypted 0 otherwise. The server then adds up these values as follows,

$$[\Gamma_{\text{sum}}] = [\sum_{i=0}^{N-1} \Phi_i] = \prod_{i=0}^{N-1} [\Phi_i] = [\sum_{i \in S} \alpha_i], \tag{5.7}$$

where Γ_{sum} is the sum of packed ratings of users in S which is the set of L most similar users. The server also computes $[L]$ that corresponds to the number of user with similarity value higher than threshold δ by simply multiplying the $[\gamma_{A,i}]$ values,

$$[L] = [\sum_{i=0}^{N-1} \gamma_{A,i}] = \prod_{i=0}^{N-1} [\gamma_{A,i}]. \tag{5.8}$$

Together with $[L]$, the server sends the $[\Gamma_{\text{sum}}]$ to user A. To obtain the final recommendations, the user A runs the secure decryption protocol with the PSP, unpacks the recommendations and divides each value by L taking into account the scaling parameter f to obtain the recommendations. These are, in fact, average scores of L users for the whole item set. This means that user A gets average ratings of all items. This step concludes our privacy-preserving protocol for recommender systems.

5.4.4 Packing Encrypted Values

In the presentation of our protocol, for the sake of simplicity we assumed that packing all values in one encryption is possible. However, this assumption is not true. For the computation of α values, the number of $v_{i,j}$ values that can be packed in one encryption is $T_1 = \lfloor \frac{n}{2k+\log(L)} \rfloor$ where n is the message space of the cryptosystem and $k + \log(L)$ is the number of bits required for each value. As a result, the number of α values that each user needs to send is $S_1 = \lceil \frac{M-R}{T_1} \rceil$. Similarly, the server needs more than one encryption for the packed representation of $[V_i']$'s. However, for blinding of Σ_A as described in the following section, we need to reserve κ bits in the most significant part to prevent an overflow. Thus, the number of σ values required in total can be given by $S_2 = \lceil \frac{N}{T_2} \rceil$ where $T_2 = \lfloor \frac{n-\kappa}{2k+\log(R)+2} \rfloor$. This also gives us the number of $[\Sigma_A]$ values.

5.5 Determining the first L users with highest similarity

In Sect. 5.4.2, the server computes an encryption of the packed similarity values of a user A: $[\Sigma_A] = [\text{sim}_{A,0}|\text{sim}_{A,1}| \dots |\text{sim}_{A,N-1}]$. We need to determine the similarity values in $[\Sigma_A]$ that exceed a public threshold δ. In this section we introduce a cryptographic protocol for performing these comparisons. It is based on [7, 9] which compares two encrypted values, however, the solution is modified using the packing idea in order to improve the efficiency of the protocol, both regarding computation and communication costs.

The desired outcome here is a vector of encrypted bits, $[\Gamma_A] = ([\gamma_{A,0}], \dots, [\gamma_{A,N-1}])$, where $\gamma_{A,j}$ is 1 if and only if the $\text{sim}_{A,j}$ is above the public threshold δ and 0 otherwise. Each similarity value $\text{sim}_{i,j}$ is of size $\ell = 2k + \log(R)$ bits (and so is δ). By the construction of the packed representation of 5.4.2, each of them is stored in the middle of an $(\ell + 2)$-bit "compartment," with the top and bottom bits set to 0.

5.5.1 Comparison

Focusing on a single comparison, the idea behind the present comparison protocol is similar to that of many previous ones: compute an encryption $[\tilde{d}^{(i)}] = [2^\ell + \text{sim}_{A,i} - \delta]$ and determine the most significant bit, $\tilde{d}_\ell^{(i)}$. This value will be 1 exactly when the similarity value matches or exceeds the threshold; naturally it must also remain secret.

Note that the computation of the encrypted $\tilde{d}^{(i)}$ can be performed in parallel on all similarity values in a single encryption. The server simply computes

$$[D] = \left[\sum_{i=0}^{N-1} \left(2^{\ell+2}\right)^i \cdot 2\tilde{d}^{(i)} \right] = [\Sigma_A] \cdot \left[\sum_{i=0}^{N-1} \left(2^{\ell+2}\right)^i \cdot 2(2^\ell - \delta) \right], \tag{5.9}$$

which is a packed list of the $\tilde{d}^{(i)}$. Each of them are at most $\ell + 1$ bits long and therefore fit in a single compartment, as this was constructed with one bit of headroom, i.e. an additional bit set to zero. We denote the $(\ell{+}2)$-bit value of the entire i'th compartment $d^{(i)}$,

$$d^{(i)} = 2\tilde{d}^{(i)} = 2^{\ell+1} + 2\text{sim}_{A,i} - 2\delta,$$

and specify the desired outcome as the $d^{(i)}_{\ell+1}$.

The server blinds $[D]$ – and thus the $d^{(i)}$ contained therein – by adding a uniformly distributed random $(\kappa + (\ell + 2)N)$-bit number r:

$$[z] := [D] \cdot [r] = [D + r]. \tag{5.10}$$

Note that as (5.9) and (5.10) both add a known value, for efficiency they should of course be implemented as a single multiplication. The server then computes $r^{(i)}$ such that $r \bmod 2^{N(\ell+2)} = \sum_{i=0}^{N-1} r^{(i)}(2^{\ell+2})^i$; each $r^{(i)}$ corresponds to the masking value for the compartment i. Notice that similarly to the $d^{(i)}$, the integers $r^{(i)}$ are $\ell + 2$ bits long. At this point the server rerandomizes $[z]$ and sends it to the PSP.

The PSP decrypts the received $[z]$ and computes each $z^{(i)}$ values such that $z \bmod 2^{N(\ell+2)} = \sum_{i=0}^{N-1} z^{(i)}(2^{\ell+2})^i$, i.e. the unpacking of z in the plain domain. At this point each of the N compartments have been separated into values, $r^{(i)}$ and $z^{(i)}$. Further, $z^{(i)} = r^{(i)} + d^{(i)} \bmod 2^{\ell+2}$ *except* that carries may propagate from one compartment to the next.

As $z = D + r$, it is clear that for every bit-position, j, $z_j = D_j \oplus r_j \oplus C_j$, where C_j is the j'th carry-bit of the addition of D and r. Hence, we have $d^{(i)}_{\ell+1} = r^{(i)}_{\ell+1} \oplus z^{(i)}_{\ell+1} \oplus C_{i(\ell+2)+(\ell+1)}$. If additive sharings of these N carry-bits modulo 2 were given (i.e. if the PSP knew $C^{\text{PSP}}_{i(\ell+2)+(\ell+1)}$ and the server knew $C^{\text{server}}_{i(\ell+2)+(\ell+1)}$, such that $C_{i(\ell+2)+(\ell+1)} = C^{\text{PSP}}_{i(\ell+2)+(\ell+1)} \oplus C^{\text{server}}_{i(\ell+2)+(\ell+1)}$), then for each similarity value, the PSP could simply encrypt $d^{(i,PSP)}_{\ell+1} = z^{(i)}_{\ell+1} \oplus C^{\text{PSP}}_{i(\ell+2)+(\ell+1)}$ and send it to the server.

The server could then compute the encryptions of the $d^{(i)}_{\ell+1}$ by either retaining $[d^{(i,PSP)}_{\ell+1}]$ or flipping it (computing $[1] \cdot [d^{(i,PSP)}_{\ell+1}]^{-1} = [1 - d^{(i,PSP)}_{\ell+1}]$) depending on the value of $r^{(i)}_{\ell+1} \oplus C^{\text{server}}_{i(\ell+2)+(\ell+1)}$.

For each i, $0 \leq i < N$, the server and the PSP determine the desired carry-bits by running the protocol summarized in the following section. Assuming that these bits are correctly computed, the server at this point has N encryptions, $[d^{(i)}_{\ell+1}] = [\gamma_{A,i}]$. So after these N executions, the vector $[\Gamma_A]$ has been obtained.

5.5.2 Obtaining the $\oplus$ Sharing of the Carry-bits

In the previous section, the comparison of similarity values $\text{sim}_{i,j}$ to the threshold δ is reduced to that of obtaining N additive sharings modulo 2 of carry-bits of the addition of the secret D and the random r generated by the server. Each of these is computed by running a comparison protocol where the server knows one input and the PSP the other. This subprotocol is adapted from [7, 9].

The key observation is that if the desired carry-bit of compartment i is set, then $r^{(i)} \bmod 2^{\ell+1} > z^{(i)} \bmod 2^{\ell+1}$. Clearly the sum (modulo $2^{\ell+1}$) of $d^{(i)} \bmod 2^{\ell+1}$

and $r^{(i)} \bmod 2^{\ell+1}$ is bigger than $r^{(i)} \bmod 2^{\ell+1}$ except if an overflow occurred. An overflow from the compartment below will not change this. As $d^{(i)} = 2\tilde{d}^{(i)}$, it is guaranteed that $d_0^{(i)} = 0$. Hence, a propagated carry may simply be viewed as "part of $d^{(i)}$" in the above intuition.

At this point all that is needed is a comparison of each of the $r^{(i)} \bmod 2^{\ell+1}$ and $z^{(i)} \bmod 2^{\ell+1}$ where the outcome is $\oplus$ shared. This happens just before termination in the comparison of [9]. We list the overall steps performed, for a full description see the original paper.

1. The PSP sends DGK encryptions of the bits of each of the $z^{(i)} \bmod 2^{\ell+1}$; $N(\ell+1)$ encryptions in all.
2. Based on the bits of the $r^{(i)} \bmod 2^{\ell+1}$, the server computes encryptions containing only a masking of the desired result; these are then sent to the PSP. The main idea here is that the server picks uniformly random bits, $b^{(i)}$, and specifies the goal as either $r^{(i)} \geq z^{(i)}$ or $z^{(i)} \geq r^{(i)}$ depending on these.
3. The PSP decrypts and determines the comparison results. However, as it does not know the $b^{(i)}$ – i.e. the direction of the comparisons – no information is revealed (the $b^{(i)}$'s can be viewed as onetime pads).

This concludes the computation, as the $b^{(i)}$'s and the outcomes of the comparisons are exactly $\oplus$ sharings of the results. The PSP and the server then use the shared results they have obtained here to provide the server with an encryption of each of the results the comparisons of the $\text{sim}_{i,j}$ and δ as described above. Correctness of the entire protocol follows by the discussion underway.

5.6 Security Analysis

We assume that all participants in the recommender system, including users, server and PSP, are honest but curious. They all follow the rules of the protocol, but will collect all their information and try to compute private information from this. We assume that the server and PSP do not collude – procedural, organizational or legal steps should be taken to ensure this. Both may, however, collaborate with users, potentially including A. We only consider static attackers meaning that the set of corrupt parties must be specified in advance. It does not appear that adaptive adversaries have any advantage; the restriction is required for technical reasons during the proof.

Intuitively, the inputs of honest parties and A's output are hidden from any attacker. The server only sees encryptions of the inputs under the PSP's public key, while the PSP only receives (encryptions of) masked values from which no information can be learned. More formally, security is shown by defining an ideal functionality and providing a simulator argument: the view of the corrupt parties (inputs, randomness, and messages) can be simulated (an indistinguishable view can be generated in polynomial time), implying that any attack against the protocol also works in an ideal setting. The reader is referred to Goldreich [10] for the full, formal definition.

The desired ideal functionality, $\mathcal{F}_{\text{RS}}$, simply receives the inputs from all parties, i.e. a preference vector $V_i = (v_{i,0}, \ldots, v_{i,M-1})$ from each user i, and the query of

A. The server and PSP are merely facilitators "authorizing" the computation but providing no inputs themselves. $\mathcal{F}_{\text{RS}}$ then determines the output $f_A = (L, \Gamma_{\text{sum}})$ – the number of similar users and the sum of their ratings – as specified above. This is sent to A, while all other parties receive empty outputs.

Users. With the exception of A, the entire view of any user i, $0 \leq i < N$ consists only of its input V_i and the encryption of that input. They receive no messages, hence perfectly simulating the entire view is trivial. It is merely the input and its encryption under the (simulated) public key of the PSP. The case of multiple users is analogous.

Server. The view of the server contains the encrypted inputs of all users $0 \leq i < N$, $[V_i'] = ([v_{i,0}'], \ldots, [v_{i,R-1}'], [\Lambda_i])$, where $v_{i,j}'$ is the scaled version of $v_{i,j}$, and Λ_i is the packed representation of the remaining preference elements. In addition to this, it consists of the messages received underway, e.g. during secure multiplications and comparisons. However, all these intermediate values are simply encryptions under one of the PSP's public keys.

As the entire view of the server consists only of encryptions, and both the Paillier and DGK encryptions schemes are semantically secure, all messages from the PSP and the honest users can therefore be simulated with encryptions of 0 under the relevant key. This is indistinguishable from the real encryptions. For the corrupt users, the inputs are known, implying that they can be simulated perfectly. It is therefore impossible for the server to extract any private information, even when colluding with a subset of the users (other than A).

PSP. The initial state of the PSP consists of its public and private key pair. Because of the ability to decrypt, all messages sent to the PSP during the protocol are blinded. Note that the present security argument is independent of any corrupt users.

During the secure multiplication protocol, the server adds random values that are κ (the security parameter) bits longer than the actual ones. These sums are indistinguishable (except with probability negligible in κ) from random values of the same length as the masks. Hence it can be simulated by providing the PSP with fresh encryptions of such random values. When determining the L users with highest similarity, the PSP obtains $[\Sigma_A + r]$, which also reveal no information by the same argument. This is also the case when running the secure decryption protocol at the end.

Security of the comparison protocol is analogous. During each invocation, the PSP obtains $\ell + 1$ DGK encrypted values. Each such tuple contains a blinded bit t_i. The blinding is more complicated than above, but the outcome is the same: the messages received can be simulated by providing fresh encryptions of random values from a specified distribution. See [7, 9] for details.

User A. It remains to argue that the view of user A does not compromise security, even if the server or PSP are also corrupt. Until the execution of the final secure decryption protocol with the PSP, A is no different from any other user, except for the role of the input in the secure computation. A's view consists only of its input and the encryption it should generate, which is easily simulated.

In the concluding decryption protocol, A should know its random mask r (selected from its own randomness) and receive first a fresh encryption of f_A from the server and then the decrypted $r + f_A$. This is easily simulated, as the simulator knows both the randomness and f_A. If the server and A collude, then the server must receive this encryption before passing it on. If A colludes with the PSP, its view must also be altered slightly. The simulator must provide an encryption of the correctly masked value in the decryption protocol rather than simply a random one.

If A colludes with *many* other users, then clearly combining their inputs with the outcome will reveal some information on the inputs of the remaining ones. E.g. if all but one users are corrupt, leaking whether the final user is similar to A is unavoidable. This is a property of the protocol, and *not* a security issue. The desired goal is for A to obtain the recommendations, and these *are* related to the inputs of other users. In practice, though, it is unlikely that any significant subset of the parties will collude in a large scale setting.

5.7 Performance Analysis

The performance analysis of our protocol is mainly determined by the interaction between the server and the PSP. The users are only participating in the protocol in two stages: 1) when they first enter the system and upload their encrypted data and 2) when they receive the encrypted recommendation. Thus, the whole workload of the protocol is shared between the server and the PSP.

Round Complexity. The round complexity of our protocol is constant and 6 rounds. The data transfer from users to the server in the initialization stage is 0.5 round. To determine the similar users and generating the recommendation, the server and the PSP need 4 rounds of interaction. Notice that during the comparison protocol to obtain $[\Gamma_A]$, all encrypted values are compared to a public value δ and, all comparisons can be done in parallel. In the last stage, the server sends the recommendation to the user which requires another 0.5 round and the user runs a protocol together with the PSP for the secure decryption protocol which is 1 round. This gives $\mathcal{O}(1)$ rounds.

Communication Complexity. The amount of data transferred during the protocol is primarily influenced by the size of the encrypted data. For a single user, the amount of encrypted data to be transferred is $\mathcal{O}(R + S_1)$. The server, on the other hand, has to receive and send $\mathcal{O}(N(R + S_1 + \ell) + RS_2)$ encrypted data which is heavily influenced by the data transmission from all N users during the initialization. The PSP has a communication complexity of $\mathcal{O}(N(S_1 + \ell) + RS_2)$.

Computation Complexity. The computational complexity is dependent on the cost of operations in the encrypted domain and can be categorized into four classes: encryptions, decryptions, multiplications and exponentiations. In Tables 5.1 and 5.2, we provide the average numbers for each operation in the Paillier and the DGK cryptosystem, respectively. One exception is for the decryption operation, which is actually a *zero-check* which is a fast and less expensive operation compared to original decryption in DGK cryptosystem.

Optimizations. The heavy operation of packing user data by the server is repeated for every user who requests a recommendation. Notice that the content of the packed data only differs for one user who is the request owner. As an improvement, the server can pack the entire users data only once and use it whenever needed. In that case, the Σ_A will contain the similarity value for user A himself. The problem of finding himself as the most similar user can be eliminated in the consequent step where

Table 5.1: Computational Complexity (Paillier).

	Server	PSP	User
Encryption	$\mathcal{O}(NS_1 + RS_2)$	$\mathcal{O}(NS_1 + RS_2)$	$\mathcal{O}(R + S_1)$
Decryption	-	$\mathcal{O}(NS_1 + RS_2)$	-
Multiplication	$\mathcal{O}(NS_1 + RS_2)$	-	$\mathcal{O}(1)$
Exponentiation	$\mathcal{O}(NS_1 + RS_2)$	-	-

Table 5.2: Computational Complexity (DGK).

	Server	PSP	User
Encryption	$\mathcal{O}(N\ell)$	$\mathcal{O}(N\ell)$	-
Decryption	-	$\mathcal{O}(N\ell)$	-
Multiplication	$\mathcal{O}(N\ell^2)$	-	-
Exponentiation	$\mathcal{O}(N\ell)$	-	-

similarity values above a threshold is found. As the position of the user is known, the similarity value at this position can be omitted. In overall, this modification will introduce substantial improvement in terms of computation cost.

It is also important to note that the complexity analysis of our protocol shows the numbers assuming that all similarity values for N users are computed each time. In general, where N is in the order of millions, this is not the case. A much smaller subset of N can be selected at random and recommendations can be generated by using this subset. Finally, assuming that the server and the PSP are separate entities with high computation power, it is realistic to anticipate a reasonable run time of generating recommendations for users.

5.8 Conclusion

We proposed a cryptographic method that eliminates threats against the user privacy in recommendation systems. The service provider in our construction consists of a server and a PSP. While the PSP possesses the decryption key, the server stores the encrypted data, meaning neither of them has access to the user data directly. In our protocol the server and the PSP can generate recommendations for the users without obtaining any information on the input, the intermediate values or the output of the algorithm. This provides full privacy for the user. The cost of processing encrypted data that is encrypted with an asymmetric cryptosystem, on the other hand, is also reduced significantly by packing user data and the intermediate values of the algorithm when possible. The performance analysis of the entire protocol shows that our protocol is promising to be deployed in real systems.

References

[1] Internet usage statistics. `http://www.internetworldstats.com/stats.htm`, 2009.

[2] List of social networking websites. `http://en.wikipedia.org/wiki/List_of_social_networking_websites`, 2009.

[3] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, 2005.

[4] J. F. Canny. Collaborative filtering with privacy. In *IEEE Symposium on Security and Privacy*, pages 45–57, 2002.

[5] J. F. Canny. Collaborative filtering with privacy via factor analysis. In *SIGIR*, pages 238–245, New York, NY, USA, 2002. ACM Press.

[6] R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty computation from threshold homomorphic encryption. In *EUROCRYPT '01: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques*, pages 280–299, London, UK, 2001. Springer-Verlag.

[7] I. Damgård, M. Geisler, and M. Krøigaard. Efficient and Secure Comparison for On-Line Auctions. In J. Pieprzyk, H. Ghodosi, and E. Dawson, editors, *Australasian Conference on Information Security and Privacy — ACSIP 2007*, volume 4586 of *LNCS*, pages 416–430. Springer, July 2-4, 2007.

[8] I. Damgård, M. Geisler, and M. Krøigaard. A correction to "Efficient and secure comparison for on-line auctions". Cryptology ePrint Archive, Report 2008/321, 2008. `http://eprint.iacr.org/`.

[9] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, R. L. Lagendijk, and T. Toft. Privacy-preserving face recognition. In *Proceedings of the Privacy Enhancing Technologies Symposium*, pages 235–253, Seattle, USA, 2009.

[10] O. Goldreich. *Foundations of Cryptography. Basic Applications*, volume 2. Cambridge University Press, first edition, May 2004. ISBN 0-521-83084-2.

[11] O. Goldreich, S. Micali, and A. Wigderson. How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority. In *ACM Symposium on Theory of Computing — STOC '87*, pages 218–229. ACM, May 25-27, 1987.

[12] P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In J. Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *LNCS*, pages 223–238. Springer, May 2-6, 1999.

[13] H. Polat and W. Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *ICDM*, pages 625–628, 2003.

[14] H. Polat and W. Du. SVD-based collaborative filtering with privacy. In *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*, pages 791–795, New York, NY, USA, 2005. ACM Press.

[15] J. R. Troncoso-Pastoriza, S. Katzenbeisser, M. U. Celik, and A. N. Lemma. A secure multidimensional point inclusion protocol. In *ACM Workshop on Multimedia and Security*, pages 109–120, 2007.

[16] S. Zhang, J. Ford, and F. Makedon. Deriving private information from randomly perturbed ratings. In *Proceedings of the Sixth SIAM International Conference on Data Mining*, pages 59–69, 2006.

Six
Anonymous Fingerprinting

This chapter has been published as “Anonymous Fingerprinting with Robust QIM Watermarking Techniques”, by J. P. Prins, Z. Erkin, and R. L. Lagendijk in the *Eurasip Journal on Information Security* 2007.

Abstract

Fingerprinting is an essential tool to shun legal buyers of digital content from illegal redistribution. In fingerprinting schemes, the merchant embeds the buyer's identity as a watermark into the content, so that the merchant can retrieve the buyer's identity when he encounters a redistributed copy. To prevent the merchant from dishonestly embedding the buyer's identity multiple times, it is essential for the fingerprinting scheme to be anonymous. Kuribayashi and Tanaka [8] proposed an anonymous fingerprinting scheme based on a homomorphic additive encryption scheme, which uses basic quantization index modulation (QIM) for embedding. In order for this scheme to provide sufficient security to the merchant, the buyer must be unable to remove the fingerprint without significantly degrading the purchased digital content. Unfortunately, QIM watermarks can be removed by simple attacks like amplitude scaling. Furthermore the embedding positions can be retrieved by a single buyer, allowing for a locally targeted attack.

In this paper, we use robust watermarking techniques within the anonymous fingerprinting approach proposed by Kuribayashi and Tanaka. We show that the properties of an additive homomorphic cryptosystem allow for creating anonymous fingerprinting schemes based on distortion compensated QIM (DC-QIM) and rational dither modulation (RDM), improving the robustness of the embedded fingerprints. We evaluate the performance of the proposed anonymous fingerprinting schemes under additive noise and amplitude scaling attacks.

6.1 Introduction

Intellectual property protection is a severe problem in today's digital world, due to the ease of illegal redistribution through the Internet. As a countermeasure to deter people from illegally redistributing digital content such as audio, images and video, a fingerprinting scheme embeds specific information related to the identity of the buyer by using watermarking techniques. In conventional fingerprinting schemes, this identity information is embedded into the digital data by the merchant and the fingerprinted copy is given to the buyer. When the merchant encounters redistributed copies of this fingerprinted content, he can retrieve the identity information of the buyer who (illegally) redistributed his copy. From the buyer's point of view, however, this scenario is unattractive because during the embedding procedure, the merchant obtains the identity information of the buyer. This enables a cheating merchant to embed the identity information of the buyer into any content without the buyer's consent and subsequently accuse the buyer of illegal redistribution.

To protect the identity of the buyer, *anonymous fingerprinting* schemes have been proposed [9, 15]. In [15], the buyer and the merchant follow an interactive embedding protocol in which the identity information of the buyer remains unknown to the merchant. When the buyer wishes to purchase, for instance, an image, he registers himself to a registration centre and receives a proof of his identity with a signature of the registration centre. Then, the buyer encrypts his identity and sends both encrypted identity and the proof of identity to the merchant. The merchant checks the validity of the signature by using the public key of the registration centre. After the buyer convinces the merchant – through the provided identity proof – that the encrypted identity indeed contains the identity information of the buyer, the merchant embeds the identity information of the buyer into the (encrypted) image data by exploiting the homomorphic property of the cryptosystem. Then, the encrypted fingerprinted image is sent to the buyer, for decryption and future use.

In this scheme, the merchant can only retrieve the identity information of the buyer when it is detected in a copy of the fingerprinted image. This idea, first presented in [15], was constructed in [13, 14] using digital coins. In order to embed the identity information of the buyer, a single bit commitment scheme with exclusive-or homomorphism is used that allows for computing the encrypted XOR of two bits by multiplying their cipher-texts. In [8], Kuribayashi and Tanaka observe that this construction is not efficient because of the low enciphering rate. The single bit commitment scheme can only contain one bit of information for a $\log_2 n$-bit cipher-text where n is a product of two large primes.

In order to increase the enciphering rate, Kuribayashi and Tanaka suggested using a cryptosystem with a larger message space. They introduced an anonymous fingerprinting algorithm based on an additive homomorphic cryptosystem that allows for the addition of values in the plain-text domain by multiplying their corresponding cipher-texts. Consequently, Kuribayashi and Tanaka used a basic amplitude quantization-based scheme similar to the well-known quantization index modulation (QIM) scheme as the underlying watermarking scheme. Since QIM essentially modulates (integer-valued) quantization levels to embed information bits into a signal, QIM can elegantly be implemented in an additive homomorphic cryptosystem. However, QIM is a basic watermarking scheme that has limited robustness compared to other watermarking schemes. The embedding positions can easily be retrieved from an individual fingerprinted copy and are thus vulnerable to local attacks. Such attacks result in minimal overall signal degradation, while completely removing the fingerprint. Furthermore, QIM is vulnerable to simple, either malevolent or unintentional, global attacks such as randomization of the least significant bits, addition of noise, compression and amplitude scaling.

In this paper, we use the ideas in [8] to build anonymous versions of state-of-the-art watermarking schemes, namely *Distortion Compensated QIM* (DC-QIM) [4] and *Rational Dither Modulation* (RDM) [12]. By adapting these watermarking schemes to the anonymous fingerprinting protocol of Kuribayashi and Tanaka we improve the robustness of the embedded fingerprints and as a consequence the merchant's security. As DC-QIM and RDM are based on *Subtractive Dither QIM* (SD-QIM), they both hide the embedding locations from the buyer more effectively, preventing local, targeted attacks on the fingerprint. With respect to global attacks, like additive noise and amplitude scaling, RDM is provably equivalent in robustness, while DC-QIM is provably better in robustness against additive noise attacks. Furthermore, RDM improves the QIM scheme so that the fingerprint becomes robust to amplitude scaling attacks.

The outline of this paper is as follows. In Section 2 we introduce the basic QIM watermarking scheme, as well as the additive homomorphic cryptosystem of

Okamoto-Uchiyama [10] on which the approach in [8] is based. In Section 3, we review the anonymous fingerprinting scheme by Kuribayashi and Tanaka. In Section 4, we describe the proposed anonymous fingerprinting schemes using the subtractive dither QIM, DC-QIM and RDM watermarking schemes. Section 5 describes the experiments that evaluate the robustness of the proposed schemes compared to the original watermarking schemes. Section 6 discusses the security benefits of using specially constructed buyer id's. Conclusions are given in Section 7. A table of used symbols is provided in the Appendix A.

6.2 Watermarking and Encryption Preliminaries

6.2.1 Basic Quantization Index Modulation

Quantization Index Modulation (QIM) is a relatively recent watermarking technique [4]. It has become popular because of the high watermarking capacity and the ease of implementation. The basic quantization index modulation algorithm embeds a watermark bit w by quantizing a single signal sample x by choosing between a quantizer with even or odd values, depending on the binary value of w. These quantizers with a step size $\Delta \in \mathbb{N}$ are denoted by $Q_{\Delta-even}(\cdot)$ and $Q_{\Delta-odd}(\cdot)$, respectively.

Figure 6.1 shows the input and output characteristic of the quantizer where $w \in \{0, 1\}$ denotes the message bit that is embedded into the host data. The watermarked signal sample y then is

$$y = \begin{cases} Q_{\Delta-even}(x), & \text{if } w = 0, \\ Q_{\Delta-odd}(x), & \text{if } w = 1. \end{cases} \tag{6.1}$$

The quantizers $Q_{\Delta-even}(\cdot)$ and $Q_{\Delta-odd}(\cdot)$ are designed such that they avoid biasing the values of y, i.e. the expected (average) value of x and y are identical. The trade-off between embedding distortion and robustness of QIM against additive noise attacks is controlled by the value of Δ. The detection algorithm requantizes the received signal sample z with both $Q_{\Delta-even}(\cdot)$ and $Q_{\Delta-odd}(\cdot)$. The detected bit $\hat{w} = \{0, 1\}$ is determined by the quantized value $Q_{\Delta-even}(z)$ or $Q_{\Delta-odd}(z)$ with the smallest distance to the received sample z.

This scheme of even and odd quantizers can also be implemented by using a single quantizer with a step-size of 2Δ and subtracting/adding Δ when $w = 1$. Implementing the quantizer in this way allows for the implementation of the scheme in the encrypted domain as was shown in [8].

A serious drawback of basic QIM watermarking is its sensitivity to amplitude scaling attacks [12] in which signal samples are multiplied with a gain factor ρ. If the gain factor ρ is constant for all samples, the attack is called a fixed gain attack (FGA). In amplitude scaling attacks, the detector does not posses the factor ρ, which causes a mismatch between embedder and decoder's quantization lattices, affecting the QIM detector's performance dramatically.

Another drawback of basic QIM is that the embedding positions can be retrieved from a single copy. The embedding positions are those signal values x_i that have been (heavily) quantized to $Q_{\Delta-even}(x_i)$ and $Q_{\Delta-odd}(x_i)$, and which have a constant

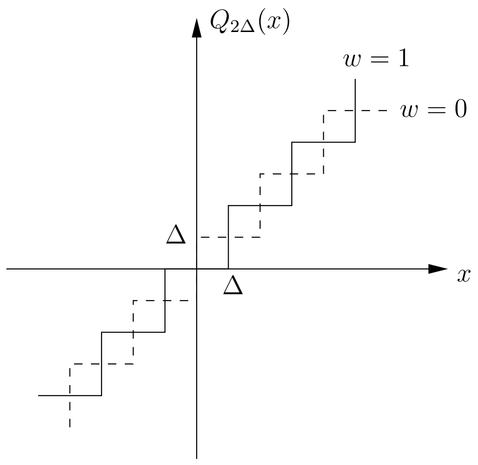


Figure 6.1: Quantizer input-output characteristic

difference value equal to Δ, i.e. the quantizer coarseness parameter. By constructing a high-resolution histogram the buyer can easily observe the even-spaced spikes of signal intensity values and identify and thus attack the embedding positions locally. This results in the removal of the fingerprint with little degradation to the overall signal.

6.2.2 Homomorphic Encryption Schemes

The idea of processing encrypted data was first suggested by Ahituv, Lapid and Neumann in [1]. In their paper, the problem of decrypting data before applying arithmetic operations is addressed and a new approach is described as processing data without decrypting it first.

Succeeding works showed that some asymmetric cryptosystems preserve structure which allows for arithmetic operations to be performed on encrypted data. This structure preserving property, called *homomorphism*, comes in two main types, namely additive and multiplicative homomorphism. Using additive homomorphic cryptosystems, performing a particular operation (e.g. multiplication) with encrypted data results in the addition of the plain-texts. Similarly, using a multiplicatively homomorphic cryptosystem, multiplying cipher-texts results in the multiplication of the plain-texts. Paillier [11], Okamato-Uchiyama [10] and Goldwasser-Micali [7] are additively homomorphic cryptosystems while RSA [16] and ElGamal [6] are multiplicatively homomorphic cryptosystems.

The anonymous fingerprinting scheme proposed in [8] is based on the addition of the fingerprint to the digital data and hence, an additive cryptosystem is used. Among the candidates, the Okamoto-Uchiyama cryptosystem is chosen for efficiency considerations [8]. In the next section, the Okamoto-Uchiyama cryptosystem is described. We observe however, that the anonymous fingerprinting schemes, proposed in this paper, can easily be implemented by using other additively homomorphic cryptosystems. It is however required to have a sufficiently large message space to represent the signal samples. Further the underlying security protocols, such as the proof protocol for validating buyer's identity, must be suitable for the chosen cryptosystem.

A requirement for the cryptosystem is that it is probabilistic in order to withstand chosen plain-text attacks. Such attacks are easily performed in our scheme, because individual signal samples are usually limited in value (e.g. 8 bit). If we were to use a non-probabilistic cryptosystem, this would enable the buyer to construct a codebook of cipher-texts for all possible messages (in total $2^8 = 256$) using the public key and decrypt through this codebook. Fortunately probabilistic cryptosystems were introduced in [7], which enable the encryption of a single plain-text to n cipher-texts, where n is a security parameter related to size of the key. To which cipher-text the plain-text is encrypted is dependent on a blinding factor r, which is usually taken at random. Selecting different r's does not affect the decrypted plain-text. By having a multitude of cipher-texts for a single plain-text the size of a codebook will become $2^8 \cdot 2^n$ and thus impractically large, preventing such attacks. All the above mentioned additive homomorphic encryption schemes (Paillier, Okamoto-Uchiyama and Goldwasser-Micali) are probabilistic and hence withstand chosen plain-text attacks.

From Section 6.3 onwards we compactly denote the encryption and the decryption of a message with $E(m)$ and $D(c)$, respectively, omitting the dependency on the random factor r. In the scope of this paper, an additive homomorphic cryptosystem will be used for encrypting signal samples which do not necessarily need to be integer values. In this case, rounding to the nearest integer value precedes the encryption and thus, in this paper, $E(\cdot)$ denotes both rounding and encryption.

Okamoto-Uchiyama Cryptosystem

Okamoto and Uchiyama [10] proposed a semantically secure and probabilistic public key cryptosystem based on composite numbers. Let $n = p^2 q$, where p and q are two prime numbers of length k bits, and g be a generator such that the order of $g^{p-1} \bmod p^2$ is p. Another generator is defined as $h = g^n$. In this scheme, the public key $pk = (n, g, h, k)$ and the secret key $sk = (p, q)$.
Encryption: A message m $(0 < m < 2^{k-1})$ is encrypted as follows:

$$c = E(m, r) = g^m h^r \bmod n, \tag{6.2}$$

where r is a random number in $\mathbb{Z}_n^*$.
Decryption: Decoding the cipher-text is defined as

$$m = D(c) = \frac{L(c^{p-1} \bmod n)}{L(g^{p-1} \bmod n)} \bmod p, \tag{6.3}$$

where the function $L(\cdot)$ is

$$L(u) = \frac{u-1}{p}. \tag{6.4}$$

The Okamoto-Uchiyama cryptosystem has the additive homomorphic property such that given two encrypted messages $E(m_1, r_1)$ and $E(m_2, r_2)$, the following equality holds:

$$\begin{aligned} E(m_1, r_1) \times E(m_2, r_2) &= g^{m_1} h^{r_1} \times g^{m_2} h^{r_2} \text{ mod } n \\ &= g^{m_1+m_2} h^{r_1+r_2} \text{ mod } n \\ &= E(m_1 + m_2, r_1 + r_2). \end{aligned} \tag{6.5}$$

Here $\times$ denotes integer modulo n multiplication.

6.3 Kuribayashi and Tanaka Anonymous Fingerprinting Protocol

The fingerprinting scheme in [8] is carried out between buyer and merchant, and has as objective to anonymously embed the buyer's identity information into the merchant's data (e.g. audio, image or video signal). The buyer decomposes his l-bit identity W into bits as $W = (w_0, w_1, \ldots, w_{l-1})$. For applications such as embedding identity information in multimedia data, the value of l is typically between 32 and 128 (bits), which is sufficiently large to prevent the merchant from guessing valid buyer id's. Where necessary, we assume that the probability $P[w_j = 0]$ and $P[w_j = 1]$ are equal. After decomposition of W into individual bits, the buyer encrypts each bit with his public key using the Okamoto-Uchiyama cryptosystem, so that $E(W) = (E(w_0), E(w_1), \ldots, E(w_{l-1}))$. These encrypted values are sent to the merchant.

The merchant first quantizes the samples of the (audio, image, video) signal that the buyer wishes to obtain, using a quantizer with coarseness 2Δ, i.e. $x' = Q_{2\Delta}(x)$. Here the quantizer step size Δ is a positive integer to ensure that the quantized value can be encrypted. He then encrypts all quantized signal samples x' with the public key of the buyer, yielding $E(x')$. The merchant selects watermark embedding positions by using a unique secret key that will be used to extract the watermark from the redistributed copies. In order to embed a single bit of information w_j into one of the quantized and encrypted value $E(x')$ at a particular watermark embedding position, the merchant performs the following operation:

$$\begin{aligned} E(y) &= E(x') \times E(w_j)^{\Delta} \\ &= E(x' + w_j \Delta). \end{aligned} \tag{6.6}$$

The result is an encrypted and watermarked signal value y, as can be readily seen by the following relation:

$$\begin{aligned} D(E(y)) &= x' + w_j \Delta \\ y &= \begin{cases} Q_{2\Delta}(x), & \text{if } w_j = 0, \\ Q_{2\Delta}(x) + \Delta, & \text{if } w_j = 1. \end{cases} \end{aligned} \tag{6.7}$$

The encrypted signal – with the buyer's identity information embedded into it in the form of a watermark – is finally sent to the buyer. Obviously, only the buyer can decrypt the watermarked signal values.

In order for the system to be robust against local attacks, the relation between the buyer's identity information bits w_j and the signal values y (audio samples, image or video pixels) into which the information bits are embedded, should be kept secret from the buyer. Note that as a consequence *all* signal values x will have to be encrypted, also the ones that do not carry a bit w_j of the buyer's identity information, as so to hide these embedding positions.

Compared to the QIM scheme in Eq. (6.1), the above watermarking scheme introduces a bias, as the expected (average) value of y is $\frac{\Delta}{2}$ larger than that of x. This bias is introduced, because Δw_j is always added to the quantized signal value x' and never subtracted. In order to avoid this undesirable side effect, either the even or odd quantizer should be selected depending on the watermark bit w_j as in Eq. (6.1). However, the merchant has only the encrypted version of each watermark bit w_j, which prevents him from deciding between the two quantizers. To overcome this problem, the merchant compares the signal values x and x', and depending on the result, the encrypted value of Δw_j can be added or subtracted [8]. When x' is smaller than x, Δw_j is added, otherwise it is subtracted. This procedure now is equivalent to Eq. (6.1) and thus effectively removes the bias. As the decision is not dependent on the value of w_j, no information is leaked about the value of w_j. The resulting embedding procedure for identity information bit w_j then becomes:

$$E(y) = \begin{cases} E(x') \times E(w_j)^{\Delta}, & \text{if } x \geq Q_{2\Delta}(x), \\ E(x') \times (E(w_j)^{\Delta})^{-1}, & \text{if } x < Q_{2\Delta}(x), \end{cases} \tag{6.8}$$

where $()^{-1}$ denotes modular inverse in the cyclic group defined by the encryption scheme. When the buyer decrypts the received encrypted and watermarked signal values, he obtains the following result for the watermark embedding positions:

$$y = \begin{cases} x' + w_j\Delta, & \text{if } x \geq Q_{2\Delta}(x), \\ x' - w_j\Delta, & \text{if } x < Q_{2\Delta}(x). \end{cases} \tag{6.9}$$

For all other positions the unwatermarked and unchanged – but encrypted and therefore rounded – signal values x are transmitted.

In the above embedding protocol, we have assumed that the buyer provides encrypted values of a valid *binary* decomposition $(w_0, w_1, \dots, w_{l-1})$ of his identity information W to the merchant. Since, however, the decomposed bits of the identity information of the buyer are encrypted, the merchant can not easily check this assumption. In the original work by Kuribayashi and Tanaka [8], a registration centre is used which assures the legitimacy of the buyer. During the purchase, the merchant first confirms the identity of the buyer, and then the buyer proves the validity of the decomposed bits of his identity information by using zero-knowledge proof protocols. Since this procedure is entirely independent of the watermarking scheme, we refer for details on the identity and decomposition validation and the security of this procedure to [8], where it is given for the Okamoto-Uchiyama encryption scheme. The focus of this paper is on the application of the homomorphic embedding procedure described above to the more robust watermarking schemes of [4, 12].

6.4 Anonymous Fingerprinting Using Advanced Watermarking Schemes

From the perspective of the merchant, the embedding of the buyer's identification information must be as robust as possible in order to both withstand malicious and benign signal processing operations on the fingerprinted signal. If the buyer id embedding procedure is not robust, the buyer could remove the fingerprint either intentionally or unintentionally and as a consequence the merchant would lose his ability to trace illegally redistributed copies. The fingerprints embedded in the Kuribayashi and Tanaka (KT) anonymous fingerprinting protocol described in Section 6.3, are known to be sensitive to a number of signal processing operations, and are in fact relatively easy to remove through attacks mentioned in Section 6.2.1. We propose to increase the robustness of the Kuribayashi and Tanaka anonymous fingerprinting protocol, as perceived by the merchant, by applying their approach to two advanced quantization-based watermarking schemes, namely DC-QIM and RDM.

So far we have embedded the bits of the identity information into signal values without specifying what these signal values actually are. In the rest of this paper we will use block-DCT transform coefficients of images to embed the identity bits into. A particular block-DCT coefficient into which we embed an information bit w_j will be abstractly denoted by x_i. Of course, in actual images, x_i may be a particular DCT coefficient of a particular DCT block in the image. The relation between the bits w_j and watermark embedding positions x_i is determined by a key known only to the merchant. In practical cases of interest, the number of candidate embedding positions is in the same order as the number of signal samples, whereas the number of information bits is typically between 32 and 128. For instance, for a 1024×1024 pixels image, the maximum number of possible embedding combinations for 128 bits of information is $\binom{1024^2}{128}$, which provides enough security. In the case of embedding the bits w_j into DCT coefficients, the number of possible embedding combinations will be smaller depending on the DCT block size and the number of DCT coefficient in one block that are (perceptually and qualitatively) suitable for embedding a watermark bit into.

It is important to note that the goal for each watermarking scheme within the Kuribayashi-Tanaka protocol is to compute the encryption of watermarked coefficients y_i, while only having available the original signal values x_i, the encrypted bits $E(w_j)$ of the buyer's decomposed identity, and the public key pk of the selected additively homomorphic encryption scheme. Once the buyer identification information is correctly embedded in the encrypted domain, the encrypted coefficients (i.e. encrypted digital content) will be sent to the buyer, who can decrypt these with his private key to obtain correctly watermarked data. Since the information bits are embedded in the DCT domain, a trivial inverse DCT on the decrypted data is necessary as the last step to obtain the purchased digital image. Because this is easiest performed in the plaintext domain we leave it to the buyer to perform this inverse DCT after decryption, which is much like JPEG decompression.

6.4.1 Subtractive Dither Quantization Index Modulation

Fingerprints embedded by the basic QIM watermarking scheme used by Kuribayashi and Tanaka as described in Section 6.2.1 can be locally attacked, because the buyer can find the embedding positions x_i without checking all possible (for instance $\binom{1024^2}{128}$) combinations. A common solution to this weakness of the basic QIM watermarking scheme is to add pseudo random noise, usually called dither, to x_i before embedding an information bit w_j, and subtracting the dither after embedding. As a consequence, the quantization levels and their constant difference Δ can no longer be observed, making the separation between embedding positions x_i and non-embedding positions impossible. The resulting watermarking scheme, illustrated in Figure 6.2, is called subtractive dither QIM (SD-QIM).

In QIM terminology, a small amount of dither d_i is added prior to quantizing the signal amplitude x_i to an odd or even value depending on the information bit w_j. After quantization of $x_i + d_i$, the same amount of dither d_i is subtracted. It is desirable that the dither can be used in cooperation with the QIM's uniform quantizers $Q_{\Delta-odd}(\cdot)$ and $Q_{\Delta-even}(\cdot)$, which use a quantization step size of 2Δ, as in the basic QIM. It has been shown [17] that a suitable choice for the PDF of the random dither d_i is a uniform distribution on $[-\Delta, \Delta]$.

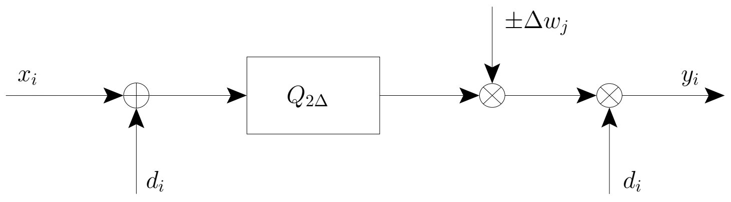


Figure 6.2: Subtractive Dither QIM

In order to embed the buyer's identity information bit $E(w_j)$ into coefficient x_i using the Kuribayashi-Tanaka protocol in combination with subtractive dither, we carry out the following protocol.

1. Add random dither d_i to the signal sample or coefficient x_i.

2. Quantize $x_i + d_i$ with a quantization coarseness of 2Δ, and encrypt the result using the buyer's public key, yielding $E(Q_{2\Delta}(x_i + d_i))$.

3. Multiply with $E(w_j)^\Delta$ or its modular inverse depending on the value of $x_i + d_i$, in order to achieve the desired quantization level.

4. Encrypt the dither d_i to obtain $E(d_i)$. Note that since $d_i \in \mathbb{R}$, the encryption operation includes modulo n rounding to an integer. Multiply the result of the previous step with the modular inverse of $E(d_i)$ as so to implement the subtraction of the dither d_i from $Q_{2\Delta}(x_i + d_i)$.

Summarizing the above protocol steps, we obtain:

$$E(t_i) = \begin{cases} E(Q_{2\Delta}(x_i + d_i)) \times E(w_j)^{\Delta}, & \text{if } x_i \geq Q_{2\Delta}(x_i), \\ E(Q_{2\Delta}(x_i + d_i)) \times (E(w_j)^{\Delta})^{-1}, & \text{if } x_i < Q_{2\Delta}(x_i), \end{cases}$$
$$E(y_i) = E(t_i) \times E(d_i)^{-1}. \tag{6.10}$$

After decryption, the buyer obtains the (DCT transformed) image into which his identity information is embedded in certain DCT coefficients y_i according to the following subtractive dither QIM scheme:

$$y_i = \begin{cases} Q_{\Delta-even}(x_i + d_i) - d_i, & \text{if } w_j = 0, \\ Q_{\Delta-odd}(x_i + d_i) - d_i, & \text{if } w_j = 1. \end{cases} \tag{6.11}$$

The above embedding procedure demonstrates the usage of the Kuribayashi-Tanaka protocol to subtractive dither QIM. The plain-text subtractive dither QIM and the above Kuribayashi-Tanaka subtractive dither QIM (KT SD-QIM) are equivalent except for the rounding of the dither d_i to integers before encryption. How to limit the adverse effect of integer rounding will be addressed next.

Two improvements of Eq. (6.10) are desirable. In the first place, we can subtract d_i before encrypting $Q_{2\Delta}(x_i + d_i)$. This effectively removes the last protocol step and hence eliminates an unnecessary encryption operation. The resulting scheme can then be rewritten as follows:

$$E(y_i) = \begin{cases} E(Q_{2\Delta}(x_i + d_i) - d_i) \times E(w_j)^{\Delta}, & \text{if } x_i \geq Q_{2\Delta}(x_i), \\ E(Q_{2\Delta}(x_i + d_i) - d_i) \times (E(w_j)^{\Delta})^{-1}, & \text{if } x_i < Q_{2\Delta}(x_i). \end{cases} \tag{6.12}$$

The second improvement concerns the quantization operation. The quantizer not only rounds the signal amplitudes to predetermined (not necessarily integer) quantization levels, but it must also round signal values or DCT coefficients $x_i + d_i$ to integers because of the ensuing encryption operation. If the signal values of DCT coefficients x_i are sufficiently large, using integer valued coefficients is not a restriction at all. For smaller values of x_i, however, using integer values may be too restrictive or may yield too large deviations between the results of Eqs. (6.12) and (6.11).

We propose to circumvent this problem by scaling all coefficients x_i with a constant factor c before embedding. Scaling has little effect on the en-/decryption, as long as the samples are not scaled beyond the message group size of the encryption scheme used. The message group size is, however, usually very large because of encryption security requirements (typically $> 2^{512}$). As a consequence of scaling x_i, the dither d_i and all encrypted bits $E(w_j)$ of the decomposed identity of the buyer also have to be scaled by c. We note that scaling introduces extra computation. However, the dither can be scaled and subtracted before encryption, resulting in a very small increase in complexity. The scaling of the encrypted bits $E(w_j)$ of the decomposed identity of the buyer has to be taken into account in the protocol steps, which is relatively easy since the scaling can be combined with the multiplication of w_j with Δ. The resulting embedding equation can be summarized as follows:

$$E(y_i) = \begin{cases} E(c \cdot (Q_{2\Delta}(x_i + d_i) - d_i)) \times E(w_j)^{\Delta}, & \text{if } x_i \geq Q_{2\Delta}(x_i), \\ E(c \cdot (Q_{2\Delta}(x_i + d_i) - d_i)) \times (E(w_j)^{\Delta})^{-1}, & \text{if } x_i < Q_{2\Delta}(x_i). \end{cases} \tag{6.13}$$

The scaling factor c has to be communicated to the buyer, so that the buyer can rescale the entire image after decryption to the proper (original) intensity range.

6.4.2 Distortion-Compensated QIM

Distortion-Compensated QIM (DC-QIM) [4] is an extension to the subtractive dither QIM scheme described in the previous section. Rather than directly adding dither to and quantizing of x_i, a fraction $\alpha \cdot x_i$ is used in the SD-QIM procedure. The information bits will be embedded only in the fraction $\alpha \cdot x_i$, where α lies within the range $[0, 1]$. The remaining fraction $(1 - \alpha) \cdot x_i$ is added back to the watermarked signal component $\alpha \cdot x_i$ to form the final embedded coefficient y_i. The embedder chooses an appropriate value for α depending on the desired detection performance and robustness of DC-QIM; an often selected value is as in [5]:

$$\alpha = \frac{\sigma_w^2}{\sigma_w^2 + \sigma_n^2} \tag{6.14}$$

where $\sigma_w^2 = \frac{\Delta^2}{3}$ is the variance of the watermark in the watermarked signal, and σ_n^2 is the variance of the noise or other degradation that an attacker applies in an attempt to render the watermark bits undetectable. Obviously, the standard SD-QIM scheme is optimal only if an attacker inserts little or no noise into the watermarked image since for $\sigma_n^2 \rightarrow 0$ we find $\alpha \rightarrow 1$. The difference in robustness between SD-QIM and DC-QIM becomes especially relevant if the variance of the attacker becomes large relative to σ_w^2, i.e. $\sigma_n^2 \rightarrow \sigma_w^2$.

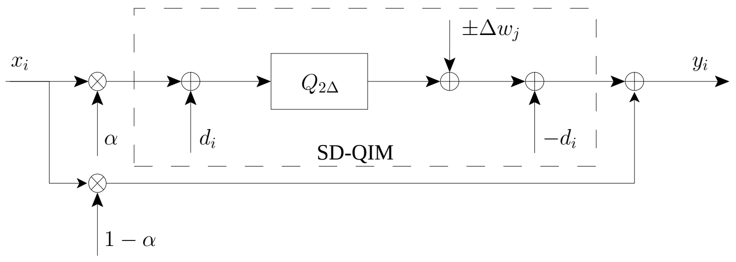


Figure 6.3: Distortion-Compensated QIM

As the differences between the SD-QIM and DC-QIM watermarking schemes merely consist of plain-text multiplications and cipher-text additions, DC-QIM can also be achieved within the limitations of the homomorphic additive encryption scheme used by the Kuribayashi-Tanaka protocol. The basic embedding operations can now be written as follows:

$$\begin{aligned}
E(t_i) &= \begin{cases} E(Q_{2\Delta}(\alpha \cdot x_i + d_i) - d_i) \times E(w_j)^{\Delta}, & \text{if } C_1 \text{ holds},\\ E(Q_{2\Delta}(\alpha \cdot x_i + d_i) - d_i) \times (E(w_j)^{\Delta})^{-1}, & \text{if } C_2 \text{ holds}.\end{cases}\\
C_1 &= \alpha \cdot x_i \geq Q_{2\Delta}(\alpha \cdot x_i),\\
C_2 &= \alpha \cdot x_i < Q_{2\Delta}(\alpha \cdot x_i),\\
E(y_i) &= E(t_i) \times E((1-\alpha) \cdot x_i). \qquad (6.15)
\end{aligned}$$

Equation (6.15) results in the following watermarked values y_i after decryption:

$$\begin{aligned}
t_i &= \begin{cases} Q_{2\Delta}(\alpha \cdot x_i + d_i) - d_i + w_j \cdot \Delta, & \text{if } \alpha \cdot x_i \geq Q_{2\Delta}(\alpha \cdot x_i),\\ Q_{2\Delta}(\alpha \cdot x_i + d_i) - d_i - w_j \cdot \Delta, & \text{if } \alpha \cdot x_i < Q_{2\Delta}(\alpha \cdot x_i),\end{cases}\\
y_i &= t_i + (1-\alpha) \cdot x_i. \qquad (6.16)
\end{aligned}$$

The plain-text distortion compensated QIM and the above Kuribayashi-Tanaka distortion compensated QIM (KT DC-QIM) are equivalent, except again for the rounding of the real valued dither d_i and $(1-\alpha) \cdot x_i$ to integers before encryption.

Similar to the subtractive dither QIM watermark algorithm, KT DC-QIM can be modified to subtract the dither before encryption, and to scale the signal values before encryption. Furthermore, the term $(1-\alpha) \cdot x_i$ can be added before encryption, further reducing the number of encryptions needed. The resulting KT DC-QIM embedding equations then become:

$$\begin{aligned}
E(t_i) &= \begin{cases} E(c \cdot (Q_{2\Delta}(\alpha \cdot x_i + d_i) - d_i)) \times E(w_j)^{\Delta}, & \text{if } C_1 \text{ holds},\\ E(c \cdot (Q_{2\Delta}(\alpha \cdot x_i + d_i) - d_i)) \times (E(w_j)^{\Delta})^{-1}, & \text{if } C_2 \text{ holds}.\end{cases}\\
C_1 &= \alpha \cdot x_i \geq Q_{2\Delta}(\alpha \cdot x_i),\\
C_2 &= \alpha \cdot x_i < Q_{2\Delta}(\alpha \cdot x_i),\\
E(y_i) &= E(t_i) \times E(c \cdot (1-\alpha) \cdot x_i). \qquad (6.17)
\end{aligned}$$

6.4.3 Rational Dither Modulation

DC-QIM provides a significant improvement in robustness compared to the basic QIM scheme. Nevertheless, the DC-QIM scheme is known to be very sensitive to gain or volumetric attacks, which is just simply scaling of the image intensities. Because of the use of the scaling factor c in SD-QIM and DC-QIM in order to reduce the sensitivity to integer-rounding before encryption, the buyer has an excellent opportunity to perform a gain attack on the watermarked signal. The gain effect causes the quantization levels used at the detector to be misaligned with those embedded in the purchased and illegally distributed digital data, effectively making the retrieval of the watermarked identity bits impossible [2].

Perez-Gonzalez *et al.* [12], proposed the usage of QIM on ratios between signal samples as so to make the watermarking system robust against fixed gain attacks. The resulting approach, known as Rational Dither Modulation (RDM), is robust against both additive noise and fixed gain attacks. The RDM embedding scheme is illustrated in Figure 6.4. The robustness against fixed gain attacks is achieved by normalizing the signal value (or DCT coefficient) x_i by $v(\mathbf{Y}_{i-1})$, which is function that combines L previous watermarked signal values $\mathbf{Y}_{i-1} = (y_{i-1}, y_{i-2}, \ldots, y_{i-L})$. An example for the function $v(\mathbf{Y}_{i-1})$ is the Hölder vector norm, as suggested in [12]:

$$v(\mathbf{Y}_{i-1}) = \left(\frac{1}{L} \sum_{m=i-L}^{i-1} |y_m|^p \right)^{1/p} \tag{6.18}$$

The SD-QIM watermark embedding will then take place using the normalized signal values $\frac{x_i}{v(\mathbf{Y}_{i-1})}$, yielding:

$$y_i = \begin{cases} v(\mathbf{Y}_{i-1}) \cdot (Q_{\Delta-even}(\frac{x_i}{v(\mathbf{Y}_{i-1})} + d_i) - d_i), & \text{if } w_j = 0, \\ v(\mathbf{Y}_{i-1}) \cdot (Q_{\Delta-odd}(\frac{x_i}{v(\mathbf{Y}_{i-1})} + d_i) - d_i), & \text{if } w_j = 1, \end{cases} \tag{6.19}$$

where the multiplication of the quantization results with $v(\mathbf{Y}_{i-1})$ is required to scale the coefficients to their original value range. Another way of viewing RDM is that it is equivalent to using SD-QIM with a signal amplitude dependent quantization coarseness $v(\mathbf{Y}_{i-1}) \cdot \Delta$.

The normalization of x_i takes place on a function of $(y_{i-1}, y_{i-2}, \ldots, y_{i-L})$ rather than of $(x_{i-1}, x_{i-2}, \ldots, x_{i-L})$. The usage of $v(\mathbf{Y}_{i-1})$ is preferable, because only the watermarked values y_i are available during watermark detection. In the Kuribayashi-Tanaka protocol the watermarked signal values or DCT coefficients y_i are only available to the merchant in an encrypted form $E(y_i)$. Unfortunately, the embedder cannot make use of $v(\mathbf{Y}_{i-1})$ as a normalization factor, primarily because homomorphic division (and multiplication for that matter) is not defined for two encrypted values in a homomorphic additive encryption scheme. Also the evaluation of the normalization function $v(\mathbf{Y}_{i-1})$ (e.g. Eq. (6.18)) may not be computable on encrypted values.

Consequently, we have to use the original signal values $(x_{i-1}, x_{i-2}, \ldots, x_{i-L})$, which will have the same statistics as $(y_{i-1}, y_{i-2}, \ldots, y_{i-L})$ for sufficiently large value of L. Experimental results have shown that an appropriate value of L is 25. For this value of L, the detection results using normalization on $v(\mathbf{X}_{i-1})$, are sufficiently close to the results based on normalization using $v(\mathbf{Y}_{i-1})$.

Since RDM applies QIM on the ratio $\frac{x_i}{v(\mathbf{X}_{i-1})}$, attention should be paid to the integer rounding process. Since $\frac{x_i}{v(\mathbf{X}_{i-1})}$ will usually be around (the real number) 1.0, the rounding to an integer will almost always yield (the integer) 1, introducing unacceptably large watermarking distortions. Therefore, the scaling of the ratio with a factor c becomes essential in RDM. Furthermore, after quantization of the ratio $\frac{x_i}{v(\mathbf{X}_{i-1})}$, the result needs to be multiplied with $v(\mathbf{X}_{i-1})$. Thanks to the homomorphic property, this can be carried out by an exponentiation in modulo arithmetic with $v(\mathbf{X}_{i-1})$ in the encrypted domain. To this end, obviously $v(\mathbf{X}_{i-1})$ has to be an integer, requiring another rounding step. In case this rounding effect is severe, another scaling can be carried out on $v(\mathbf{X}_{i-1})$. Since in our experiments this effect showed to be negligible, we do not consider scaling of $v(\mathbf{X}_{i-1})$ itself. We denote the rounded value of $v(\mathbf{X}_{i-1})$ by $v_{int}(\mathbf{X}_{i-1})$.

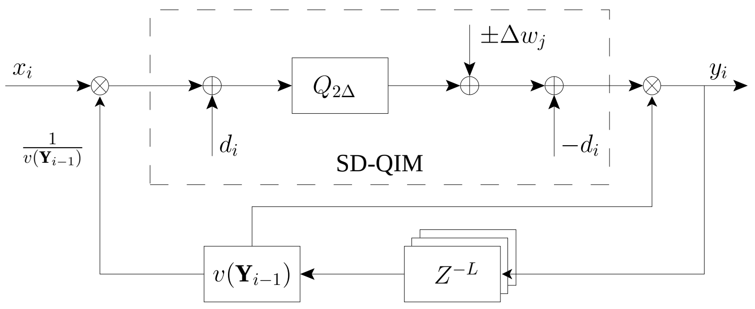


Figure 6.4: Rational Dither Modulation

Using again the notation d_i for the uniformly distributed dither, the RDM embedding equations become:

$$\begin{aligned}
E(t_i) &= \begin{cases} E(c \cdot \left(Q_{2\Delta}\left(\frac{x_i}{v_{int}(\mathbf{X}_{i-1}} + d_i\right) - d_i\right)) \times E(w_j)^{\Delta}, & \text{if } C_1 \text{ holds,} \\ E(c \cdot \left(Q_{2\Delta}\left(\frac{x_i}{v_{int}(\mathbf{X}_{i-1}} + d_i\right) - d_i\right)) \times (E(w_j)^{\Delta})^{-1}, & \text{if } C_2 \text{ holds,} \end{cases} \\
C_1 &= \left(\frac{c \cdot x_i}{v_{int}(\mathbf{X}_{i-1})}\right) \geq Q_{2\Delta}\left(\frac{c \cdot x_i}{v_{int}(\mathbf{X}_{i-1})}\right) \\
C_2 &= \left(\frac{c \cdot x_i}{v_{int}(\mathbf{X}_{i-1})}\right) < Q_{2\Delta}\left(\frac{c \cdot x_i}{v_{int}(\mathbf{X}_{i-1})}\right) \\
E(y_i) &= E(t_i)^{v_{int}(\mathbf{X}_{i-1})}. \qquad (6.20)
\end{aligned}$$

With the above scheme we have succeeded in adapting the RDM watermarking scheme – one of the most recent QIM watermarking approaches – to the constraints set by the Kuribayashi-Tanaka protocol.

6.5 Experimental Validation

In this section we experimentally compare the plain-text versions of the SD-QIM, DC-QIM and RDM watermarking schemes with the proposed version based on the Kuribayashi-Tanaka fingerprinting protocol. The buyer's identity information will be embedded into the DC DCT coefficients of 8x8 blocks. Per image we embed 64 bits of identity information into 64 DC DCT coefficients that are pseudo randomly selected based on a secret key only known to the merchant. In all experiments we use the 256×256 pixels gray-valued Lena and Baboon images. Because of runtime efficiency and the availability of the necessary proofs we selected the Okamoto-Uchiyama cryptosystem for all experiments as in [8]. The Okamoto-Uchiyama cryptosystem has a smaller encryption rate compared to (generalized versions of) Paillier, because of a smaller message space for the same security level. However as signal values are usually sampled with 8 bit precision, a smaller message space is not a problem for our application, while the cipher-text size is reduced with the Okamoto-Uchiyama cryptosystem, resulting in lower overall computational complexity.

We not only compare the performance of the plain-text and cipher-text versions of the SD-QIM, DC-QIM and RDM watermarking schemes, but we also evaluate the effect of integer rounding and the scaling parameter c on the performance. In our graphs, each point shown is based on 100 measurements, and each measurement is a complete, new iteration of the Kuribayashi-Tanaka protocol. A table of parameters[1] for algorithms can be found in the Appendix B.

6.5.1 Subtractive Dither QIM

An important performance measure of a watermarking scheme is the bit error rate (BER) of the watermark detector as a function of the strength of embedding the watermark. The BER is a measure that quantifies the probability P_e of incorrectly detecting a single bit of information. Usually, the buyer's identity information contains some form of channel coding, so that the buyer's identity can still be retrieved even if a few bits are incorrectly detected from the fingerprinted image, this is further discussed in Section 6.6.

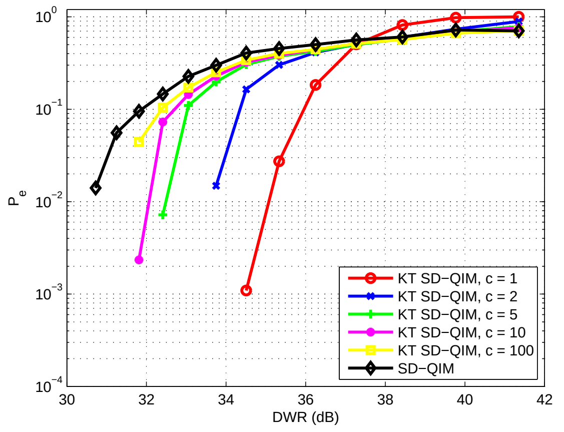
In order to measure the distortion that the watermark introduces into the host signal, we use the document-to-watermark ratio (DWR):

$$DWR = 10\log_{10}(\frac{\sigma_x^2}{\sigma_w^2}) \qquad (dB). \tag{6.21}$$

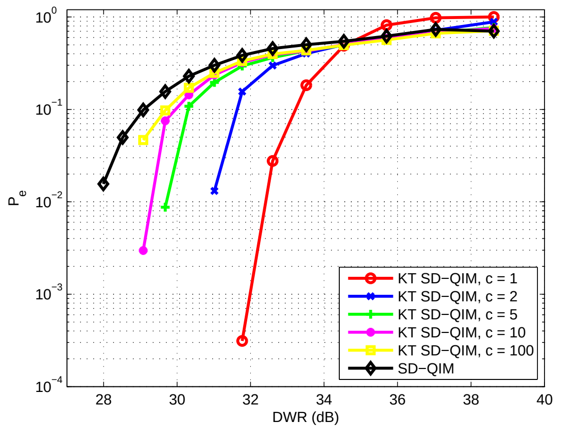
Here σ_x^2 is the variance of the data into which the watermark is embedded, which in our case are the DC DCT coefficients of 8x8 blocks. Further, σ_w^2 is the variance of the distortion caused by the embedded watermark. Following [4], we equate $\sigma_w^2 = \frac{\Delta^2}{3}$. The objective a watermarking scheme is to have a low BER with a high DWR. The proper values for the DWR and thus Δ is application and data dependent. In this paper we are not concerned with selecting a suitable value of Δ. We rather study the behaviour of the BER as a function of the DWR for the plain-text and Kuribayashi-Tanaka versions of the SD-QIM watermarking scheme.

Figure 6.5 shows the BER-DWR relation for the two versions of the SD-QIM algorithm. The performance of the Kuribayashi-Tanaka version of the SD-QIM (KT SD-QIM) watermarking scheme is shown for several values of the scaling factor c. Although there is no deliberate attack performed on the watermark, the inverse DCT transform and consequential rounding to 8 bit pixel values introduces a distortion into the fingerprinted signal. The robustness of the watermarking scheme is sufficient, however, to result in no bit errors at a DWR of 31-34 dB. A peculiar effect is the increased robustness of the heavily rounded (i.e. scaling factor $c = 1$) KT SD-QIM compared to the original watermarking scheme. We believe that this behaviour is caused by the distorting effect of the (inverse) DCT transform. By increasing the scaling factor c we can approximate the performance of the original SD-QIM. The performance is already closely approximated with c = 100 in this instance, but in general the application, the data and the implementation of the DCT will determine

[1]The codes for the implementation can be found in http://ict.ewi.tudelft.nl



(a)



(b)

Figure 6.5: SD-QIM bit error rate (BER) P_e as a function of the document-to-watermark ratio (DWR) for the original SD-QIM scheme and KT SD-QIM with different scaling factors c 1, 2, 5, 10 and 100 for a) Lena and b) Baboon images.

which value of c is required to approximate the performance of the plain-text SD-QIM scheme.

6.5.2 Distortion-Compensated QIM

Figure 6.5 showed the BER in a scenario without any explicit attacks on the watermark. Distortion-Compensated QIM can be used to provide optimal robustness against additive noise attacks. For this reason, we will show the performance of the Kuribayashi-Tanaka adaptation of DC-QIM and compare it with the original DC-QIM and the previously discussed SD-QIM. A measure of the amount of noise introduced relative to the strength of the watermark is the watermark-to-noise ratio (WNR):

$$WNR = 10 \log_{10}(\frac{\sigma_w^2}{\sigma_n^2}) \quad (dB). \tag{6.22}$$

Here σ_n^2 is the variance of the additive zero-mean Gaussian noise that the attacker adds to the fingerprinted content. The value of α is chosen according to Eq. (6.14), so that the DC-QIM scheme is tuned for a specific additive noise variance level. In all our experiments we use $\sigma_n = 15$ and change the value of $\Delta = \sqrt{3}\,\sigma_w$ as so to obtain a varying WNR.

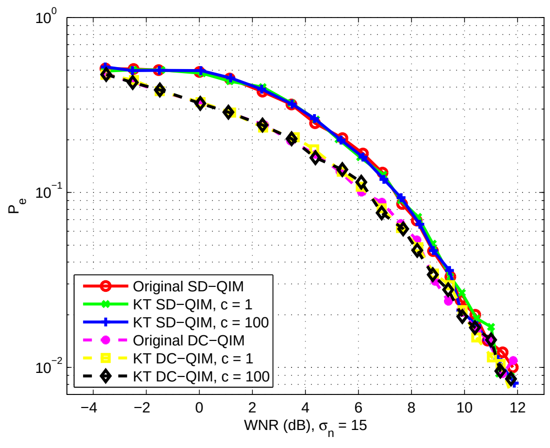
Figure 6.6 shows the BER-WNR relation for SD-QIM and DC-QIM. We choose to fix the amount of additive noise instead of the DWR, because we are interested in the effect the scaling factor c has on the required embedding strength (i.e. value of Δ and thus the watermark power) and not a variable amount of additive noise. Therefore Figure 6.6 can not be easily compared to other literature on watermark robustness. As in our previous experiment the watermark distortion is calculated using the expression $\sigma_w^2 = \frac{\Delta^2}{3}$ [4].

As can be observed the performance of the DC-QIM is better than SD-QIM with additive noise, which is in accordance with [4]. We are mostly concerned with the comparison of the original version of the DC-QIM scheme and the Kuribayashi and Tanaka adaptation of DC-QIM. As expected the performance of the original DC-QIM scheme and the Kuribayashi-Tanaka adaptation of DC-QIM (KT DC-QIM) differ very little. Also the scaling factor c has little effect on the BER. This can be explained by the fact that the additive noise dominates the errors caused by the integer rounding.

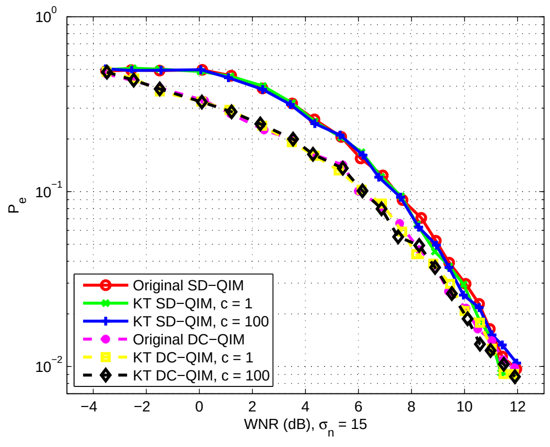
6.5.3 Rational Dither Modulation

Unlike the previous two watermarking schemes, rational dither modulation (RDM) depends on a sufficiently large scaling factor c in order to achieve a quantization coarseness Δ lower than 1. The scaling factor c determines the possible resolution of Δ. We are interested to see which resolution is required, in order to achieve good performance. Although the results depend on the data and the strength of the added noise, the trend of these results will be observed for other cases and data as well, because the signal coefficients x_i are normalized before embedding.

Figure 6.7 shows the bit error rate (BER) performance of RDM as a function of the watermark-to-noise ratio (WNR) for the plain text and Kuribayashi-Tanaka versions

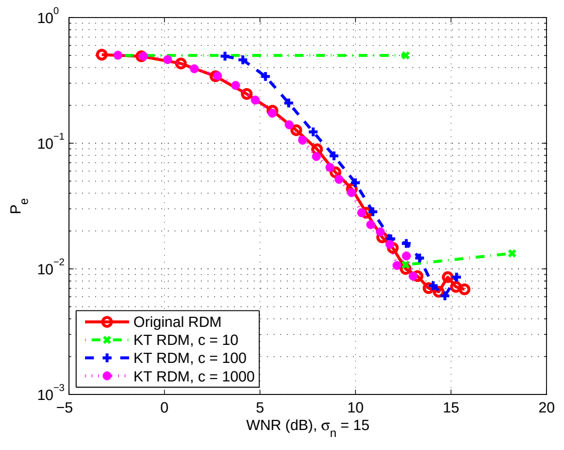


(a)

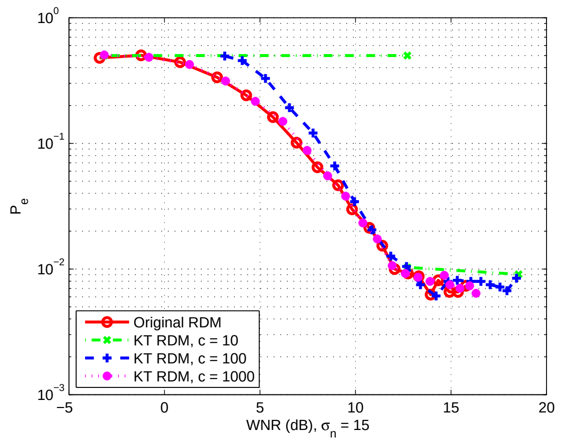


(b)

Figure 6.6: SD-QIM and DC-QIM BER as a function of WNR with additive noise ($\sigma_n = 15$) for the original SD-QIM and DC-QIM schemes and the KT SD-QIM and DC-QIM schemes with different scaling factors c for a) Lena and b) Baboon images.



(a)



(b)

Figure 6.7: RDM bit error rate (BER) as a function of the watermark-to-noise ratio (WNR) with additive noise ($\sigma_n = 15$) for the original RDM scheme and KT RDM scheme with different scaling factors c for a) Lena and b) Baboon images.

of RDM. The different curves reflect different values for the scaling factor c. Because of the complexity of the analytical expression of the watermark distortion σ_w^2 in [12], we measured the watermark distortion directly from the data.

Figure 6.7 shows that the value of the scaling factor c determines the points of the P_e-WNR curve which are attainable by the Kuribayashi-Tanaka RDM scheme. With a scaling factor $c = 10$, only WNRs with 12 dB or higher are reachable (see 'KT RDM, c = 10' curve in Figure 6.7, which starts at 12 dB), allowing for very little flexibility in choosing the optimal embedding strength for a specific application. A scaling factor of 100 performs much better, but 1000 approximates the original RDM closely.

Besides the equivalent robustness to additive noise attacks of RDM compared to SD-QIM, RDM is robust against amplitude scaling attacks. Figure 6.8 shows the robustness of SD-QIM, DC-QIM and RDM to a performed amplitude scaling attack. SD-QIM and DC-QIM show a high vulnerability against amplitude scaling attacks. At a small gain factor ρ of 1.05, approximately 50 percent of the buyer's identifying information cannot be retrieved correctly, while RDM is robust throughout the whole range for the gain factor. Although theoretically RDM should not be at all affected by an amplitude scaling attack some bit errors start to show up at gain factors larger than 1.06. These are inherent to the 8-bit data representation format, which easily overflows for large gain factors.

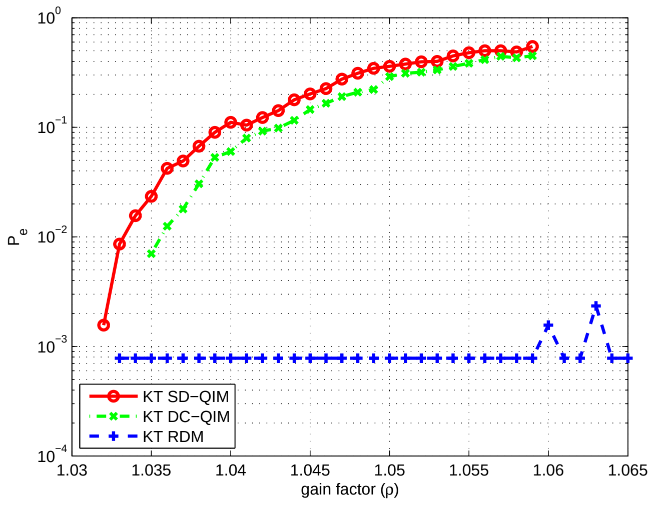


Figure 6.8: KT bit error rate (BER) as a function of the gain factor (ρ) for KT SD-QIM, KT DC-QIM and KT RDM schemes with c = 1000. The DWR is fixed to 7.1 dB. Datapoints below a BER of 10^{-3} are plotted for visualization, but in reality 0.

6.6 Security Aspects of Buyer Identity

As fingerprint detection is a signal processing operation, detected fingerprints will usually be distorted even without attacks on the fingerprint by a malicious buyer, as discussed in Section 6.4. The fingerprint can for instance be distorted by perfectly legitimate signal processing operations such as compression, the obligatory inverse DCT and consequential rounding. In this scenario the merchant would normally not be able to present a perfectly retrieved buyer id. The registration centre could accept merchant buyer id submissions, which are similar to a correct buyer id. However, the security of the buyer depends on the inability of the merchant to guess a correct buyer id. To allow the merchant to submit similar buyer id's and for the registration centre to accept these would thus harm the buyer's security.

By letting the registration center extend the buyer identity with a forward-error-correcting scheme, the merchant can compensate for a small and fixed maximum number of bit errors in the buyer id. This is of course equivalent to increasing the size of the buyer id and allowing for a small number of bit errors at the registration centre. This approach has the advantage that it moves the computational complexity of the error correction from the registration centre to the merchant.

There is a choice to be made concerning the locations of the embedding positions for each buyer. The embedding positions can be changed for each buyer, but this would not provide any real benefits to the robustness of the total fingerprinting scheme, other than that colluding buyers would have to compare their individual fingerprinted version with a number of other versions in order to detect the embedding locations. If the embedding locations are identical for each fingerprinted copy, buyers who have located these embedding positions could publish these and all buyers could then remove the fingerprint from their copy. Using unique embedding positions for each buyer has, however, a big disadvantage upon detection. As with any fingerprinting scheme, the merchant cannot know the used embedding positions before detection, as the detection procedure is the sole method to discriminate between copies. The unavailability of the embedding positions prevents the merchant from detecting the buyer id, resulting in a deadlock. In order to break this deadlock the merchant could estimate the embedding positions by using a non-blind detection procedure (e.g. subtract the original image from the encountered image and thus find the most likely candidate embedding locations, as they will be show up to have a high difference to the original signal) or by embedding a pilot signal to identify the used embedding positions. However this would be ineffective for heavily attacked copies, which are heavily distorted by attacks. Another way to retrieve the correct buyer id is to let the merchant detect for all possible embedding locations and use a (soft) error correction scheme to determine the most likely buyer id, based on the distance the detected id is from a valid codeword in the used error correction scheme. This, however, makes the detection procedure linear in complexity related to the number of buyers as it has to be performed for each used combination of embedding positions.

Although dithering prevents an individual buyer to detect the embedding positions, a coalition of buyers can collude to find them. By comparing different fingerprinted copies, the coalition can locate the differing samples/coefficients and, as the fingerprint embedding is the predominant cause of these differing samples, consequently the embedding positions. This vulnerability can be eliminated by constructing the buyer's id's through the scheme of Boneh and Shaw [3], making them collusion-secure. The collusion-security of the scheme of Boneh and Shaw depends on generating buyer id's such that they have a number of identical bits w_j for any colluding coalition of c buyers. Because these buyer id bits are identical, the coalition is not able to detect these embedded bits by comparing their individually fingerprinted copies. This does however require that the embedding positions are identical for each fingerprinted copy. Because the embedding positions for these bits cannot be determined they are safe from targeted attacks and can therefore be detected correctly by the merchant even after the attack by the colluding buyer coalition. Constructing such a collusion-secure code for a large coalition constitutes a large increase in the buyer id length. As shown in [3] the length is equal to $O(c^4 \log(N/e) \log(1/e))$, where c is the number of colluding buyers, N is the total number of buyers and e is the probability that the cheating buyer cannot be retrieved after a collusion attack. Because of the anonymity of the embedding procedure, the registration centre will have to generate the collusion-secure buyer id's, as this will be the only person the merchant trusts to generate a valid buyer id.

6.7 Conclusion

In conventional fingerprinting schemes, the buyer's identity is known to the merchant during embedding. This knowledge can be easily abused by a malicious merchant by creating fingerprinted copies containing this identity information without the buyer's consent. After distribution the merchant can claim a license violation for this specific buyer. To deal with this problem, Kuribayashi and Tanaka proposed a reasonably efficient solution in [8] based on embedding the buyer identification information using additive homomorphic encryption schemes. The problem of the proposed protocol in [8] is the vulnerability of the underlying basic QIM watermarking scheme, which is fragile to simple attacks like amplitude scaling and allows for the detection of the embedding positions. Therefore, we have proposed to adapt DC-QIM and RDM techniques to the anonymous fingerprinting scheme of Kuribayashi and Tanaka.

We have adapted DC-QIM and RDM techniques which hide the embedding locations, unlike basic QIM, because they are based on SD-QIM. They perform provably equivalent (RDM) or better (DC-QIM) than the watermarking scheme in the original work against additive noise attacks. Furthermore, RDM provides robustness to amplitude scaling attacks which is a major drawback of the basic QIM scheme used in [8].

Although rounding errors can be made arbitrarily small through the use of scaling factors, the practical need, as shown in the experiments, is small. As integer quantization step sizes have to be used because of the homomorphic encryption scheme, the distortion introduced by the fingerprint embedding is usually larger than the distortion introduced by integer rounding. As a consequence rounding with a scaling factor of one (i.e. no scaling) already has acceptable performance. The scaling factor has its use however in increasing the effective quantizer resolution. Although this is of limited use for signals with a relatively large value range, it is essential for signals with a small value range, as is the case for RDM after normalization.

Due to attacks on the digital content or transmission errors, the identity information of the buyer can be extracted with bit errors. In that case, using error correction codes can improve the abilities of the merchant to recover the identity information. By letting the registration center select the buyer identity information, we can incorporate these error correction capabilities or even provide a collusion-secure fingerprinting scheme. This greatly increases the embedded buyer's identification information and the complexity of constructing a valid identity at the registration centre. Although this might not be practical in real applications, it provides a theoretical solution to the problem of collusion.

By adapting the DC-QIM and RDM watermarking schemes to the anonymous fingerprinting protocol of Kuribayashi and Tanaka, we increased the robustness of the embedded fingerprints, while preserving the anonymity of the fingerprinting protocol. Consequently the buyer's ability to successfully attack embedded fingerprints is reduced, increasing the deterrence to the illegal redistribution of digital content.

References

[1] N. Ahituv, Y. Lapid, and S. Neumann. Processing encrypted data. *Communications of the ACM*, 30(9):777–780, 1987.

[2] F. Bartolini, M. Barni, and A. Piva. Performance analysis of ST-DM watermarking in presence of nonadditive attacks. *IEEE Transactions on Signal Processing*, 52(10):2965–2974, 2004.

[3] D. Boneh and J. Shaw. Collusion-secure fingerprinting for digital data. *IEEE Transactions on Information Theory*, 44(5):1897–1905, 1998.

[4] B. Chen and G. Wornell. Quantization index modulation: A class of provably good methods for digital watermarking and information embedding. *IEEE Transactions on Information Theory*, 47(4):1423–1443, 2001.

[5] M. Costa. Writing on dirty paper. *IEEE Transactions on Information Theory*, 29(3):439–441, 1983.

[6] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1986.

[7] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[8] M. Kuribayashi and H. Tanaka. Fingerprinting protocol for images based on additive homomorphic property. *IEEE Transactions on Image Processing*, 14(12):2129–2139, 2005.

[9] N. Memon and P. Wong. A buyer-seller watermarking protocol. *IEEE Transactions on Image Processing*, 10-4:643–649, 2001.

[10] T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Eurocrypt*, volume 1403, pages 308–318, 1998.

[11] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. *Lecture Notes in Computer Science*, 1592:223–238, 1999.

[12] F. Perez-Gonzalez, C. Mosquera, M. Barni, and A. Abrardo. Rational dither modulation: A high-rate data-hiding method invariant to gain attacks. *IEEE Transactions on Signal Processing*, 53(10 Part 2):3960–3975, 2005.

[13] B. Pfitzmann and A.-R. Sadeghi. Coin-based anonymous fingerprinting. In *Eurocrypt*, volume 1592, pages 150–164, 1999.

[14] B. Pfitzmann and A.-R. Sadeghi. Anonymous fingerprinting with direct non-repudiation. In *Asiacrypt*, volume 1976, pages 401–414, 2000.

[15] B. Pfitzmann and M. Waidner. Anonymous fingerprinting. In *Eurocrypt*, volume 1233, pages 88–102, 1997.

[16] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[17] I. D. Shterev and R. L. Lagendijk. Amplitude scale estimation for quantization-based watermarking. *IEEE Transactions on Signal Processing*, 54(11):4146–4155, 2006.

6.A Table of Parameters

Algorithm	Scaling Factor	Quantization Stepsize	Noise
SD-QIM	$c = 1, 2, 5, 10, 100$	$\Delta = k$ for k, $1 \leq k \leq 20$	
DC-QIM	$c = 1, 10, 100$	$\Delta = 5k$ for k, $1 \leq k \leq 20$	$\sigma_n = 15$
RDM	$c = 10$	$\Delta = k$ for k, $1 \leq k \leq 20$	$\sigma_n = 15$
	$c = 100$	$\Delta = k$ for k, $1 \leq k \leq 20$	
	$c = 1000$	$\Delta = 8k$ for k, $1 \leq k \leq 20$	
	$c = 10000$	$\Delta = 75k$ for k, $1 \leq k \leq 20$	

Seven

Discussion

In this thesis, we have addressed the privacy aspects of online multimedia applications which are widely used by a vast number of people. These applications, ranging from shopping to dating, present serious privacy risks since the information required from the users are highly privacy sensitive and open to misuse by the service provider itself. As a solution to the privacy threats in multimedia applications, we propose to use cryptographic techniques in the design of the multimedia applications. In particular, we promote the idea of keeping the privacy-sensitive data safe by means of encryption and processing them under encryption. The required signal processing operations can be realized in the encrypted domain by exploiting homomorphism property of certain public key cryptosystems and using MPC techniques.

In order to introduce a methodology to achieve privacy protection in multimedia applications, we selected a number of prototypical applications and presented detailed cryptographic protocols for each application that are correct, privacy-preserving and efficient in this thesis. The selected applications, namely face detection, clustering, recommender systems and anonymous fingerprinting, contain similar signal processing properties regarding the structure of the data and corresponding operations on them. In this chapter, we investigate the selected applications, deduce some conclusions and point out directions for future research.

7.1 Variations in the protocols with respect to use scenarios

As a result of our observations on the selected prototypical applications presented in this thesis, we notice that the required solutions have commonalities with respect to the signal processing operations and data structure. However, we also observed that the realization of signal processing operations in the encrypted domain may differ according to the followings:

- **Application setting.** The type and the number of parties.

  - *Two-party.* This setting consists of only two players, user and server as in face detection (Chapter 3) and anonymous fingerprinting (Chapter 6).
  - *Multi-party.* Even though there are two types of players, server and user, the number of players can vary as in clustering (Chapter 4) and recommender systems (Chapter 5) where there are vast number of users.

  The choice on the number of parties plays an important role in computation load share. In a two-party setting, the computation load is usually shared between the server and the user. In order to reduce the workload of users, a third player can be introduced to participate in the protocol. As in the recommender system in Chapter 5, a semi-trusted third party, namely PSP, participates in the protocol and the user is only required to upload his data and receive the outcome of the protocol.

  Remember that in Chapter 1, we argue that having a TTP is not a realistic assumption. However, for the privacy-enhanced recommender system, we propose to use a PSP who is semi-trusted. One should note that the PSP required for the recommender systems is not the TTP described in Chapter 1. The TTP is trusted by all such that he is given the privacy-sensitive data and the algorithm. However, the PSP is not trusted in that sense. As it can be seen in the security analysis of the system in Section 5.6, the PSP is not given any plain data but only encrypted and blinded. The PSP is trusted in the sense that he performs the operations on the encrypted data as described.

  Another important observation is that our protocol constructions require interaction with the secret key owner. This means that the service provider cannot initiate another protocol with another player without the consent of the secret key owner. Considering the example of automated medical scenario introduced in Section 1.1, the expert system cannot ask the opinion of another entity without informing the secret key owner, the patient in this case. However, note that with the involvement of the patient in the protocol, other players, i.e. other medical players, can initiate other cryptographic protocol. Actually, this is what we would like to emphasize in this thesis. Given that the privacy-sensitive data is encrypted and the owner of the secret key is involved, it can be processed under encryption. Involvement of the secret key owner is an requirement because of the technical challenges such as interactive protocols based on MPC techniques. This can be seen as an advantage since the secret key owner, the patient, is needed to be involved in the protocols to process the encrypted data. This can be considered as an effective control mechanism.

- **Owner of the decryption key.** In all the applications discussed in this thesis, the users in the applications are to be protected. However, depending on the application setting, the owner of the secret key may change. In a two-party setting, the owner of the data has the secret key as in face recognition and anonymous fingerprinting, Chapters 3 and 6. However, in a multi-party setting as in clustering and recommender systems, Chapters 4 and 5, we proposed a system where users have their own private data and the server has the secret key. This approach is preferred because if each user possesses his own secret key or a shared key to protect his private data, the privacy-preserving protocol becomes more resource demanding as this approach requires deploying other cryptographic techniques like threshold schemes which are considerably more costly compared to proposed solutions in this thesis.

7.2 Achievements

Having noted that the cryptographic solutions for preserving privacy in a multimedia application change depending on the above choices, we can now focus on the proposed solutions for the challenges defined in Section 2.5. Note that the correctness and the privacy requirements are satisfied and discussed for each application in the related chapters separately. In the followings, we provide an overall view regarding the addressed challenges and discuss their efficiency aspects.

7.2.1 Data Representation

In multimedia applications, we usually deal with values that can be classified as signals. As in the example of clustering, the users are represented in a R-dimensional space in which the magnitude in each dimension can take integer values from a small range, usually a few bits. Such a generalization can be made for many other multimedia applications since the data in question are usually in the form of signal samples such as images, preferences and feature vectors. For example, in face recognition and anonymous fingerprinting, the inputs are 8-bit gray scale images. In clustering and recommender systems, we represent each user with his preference or rating vector whose elements are only 4-bits. This observation can be seen as a significant advantage in designing privacy-preserving multimedia applications but it has two important aspects. Firstly, the assumption on the type of the values is not true. Even though the initial data might be integer values, they become real values throughout the processing. For instance, the similarity value for two users in the recommender system is a real value between -1 and 1. However, existing public key cryptosystems mostly work on integers. Secondly, the assumption on the size of the values is misguiding. Computations might begin with values in a small range but during processing values get larger. As an example, consider the Euclidean distance computation between two vectors. Although the vector elements are small values, the result of the computation is larger in bit size. Note that, truncating intermediate values during the run of the protocol is not practical since it requires interaction and computationally expensive protocols for operations like division.

In this thesis, to cope with the non-integer form of values, we propose to scale and round values with enough precision before encryption. The required precision can be easily computed by analyzing the operations. In face recognition, for instance, we propose to use a scaling factor of 1000 which is sufficient for the correctness of the application. However, note that the scaling parameter will directly affect the cryptographic protocols for the consequent steps. As discussed in Sections 3.5, 4.3 and 5.5, the cryptographic protocols work on the actual bit length of the values to be compared. With longer bit length, it takes more time to finish the cryptographic comparison protocol. Thus, the length of the scaling factor plays an important role.

To eliminate the problems that may occur due to the increase in bit length of the values, we propose to reserve sufficient space for the consequent computation steps. For instance, in recommender systems, two k-bit user ratings are to be multiplied and such R values are to be added. Therefore, we reserve $2k + \log(R)$ bits for each value (Section 5.4.1). Reserving sufficient space is especially important in the case of data packing. As the signal values are packed one after another in one encryption and going to be processed together, reserving enough space is necessary to guarantee the correctness of the operations in the encrypted domain. Each value should be placed in compartments with a suitable size so that the expansion due to consequent processing steps does not alter the other compartments. The size of the compartment for each value can be calculated by analyzing the operations. Similar to scaling, the size of the compartments effects the cryptographic protocols. Thus, the size of the compartments should be determined by analyzing the required precision and operations.

7.2.2 Linear Operations and Homomorphism

Additive homomorphism plays a crucial role in processing encrypted data as it allows adding and scaling values in the encrypted domain. This property is used frequently in all of the applications covered in this thesis. In order to explain the limitations of homomorphism, we can investigate the linear signal processing operations covered in this thesis.

The homomorphic encryption can be used to calculate distances (squared Euclidean and Hamming) and correlation between two vectors. However, it depends on application setting whether the whole computation can be done only by using homomorphism or not. As in clustering and recommender systems, distance computation and Pearson correlation can be computed by one of the parties using homomorphism (Sections 4.2.1 and 5.4.2). In both cases, each party has its own private vector. The correlation is computed by one of the parties upon receiving the encrypted vector from the other. On the other hand, as in the case of face detection, a similar computation, squared Euclidean distance, requires running a secure multiplication protocol for the computation of the product terms since the vectors to be processed are both private and possessed by the same party while the secret key is possessed by the other (Section 3.4.2). These two different solutions for the computation of a similar signal processing operation show that homomorphism is required but may not be sufficient depending on the setting.

As another linear operation on encrypted data, it is also important to pay attention to blinding. While semantic security protects the content of encryption, it is not sufficient to protect the data when the same party holds the decryption key as seen in secure clustering (Section 4.2.3). In such cases, blinding, also known as masking, plays an important role.

In face detection and recommender systems, two secure multiplication protocols are needed (Sections 3.4.2 and 5.3.2). In face recognition, the server has an encrypted value that he wants to keep it secret from the user and needs to square it. In recommender system, the server has two encrypted values that he wants to keep secret from the PSP and needs to multiply them. In both cases, the procedure is the same. The values are first blinded by adding a random value and sent to the other party who has the decryption key. The blinded values are then decrypted, multiplied (or squared) and encrypted. Upon receiving the encryption, the blinding factor is removed to obtain the actual outcome of the multiplication.

Blinding also works in other key situations. In clustering, for instance, all users need to send their encrypted data to the server to be accumulated (Section 4.2.3). However, the server possesses the secret key and hence, he can observe the contents of the encryptions that are sent by each user. To prevent the server from accessing the users' contribution for updating the centroids, each user blinds their contribution in such a way that only when all contributions are added up, the server can obtain the accumulated result. Therefore, blinding demonstrates a simple but effective technique of hiding private data. Similarly, in recommender systems, the user obtains the recommendations by running a secure decryption protocol with the PSP in which the content of the encryptions are kept secret by blinding the content (Section 5.3.3).

7.2.3 Non-Linear Operations and MPC

Linear operations are important for signal processing applications; however, they constitute only a part of the processing. For instance, distance computations are often followed by a decision like selecting the minimum distance. In order to realize such non-linear operations in the encrypted domain, we need to exploit protocols based on MPC techniques.

The prototypical applications selected for this thesis show that a crucial operation for signal processing applications is comparison which takes two encrypted values and outputs the comparison result encrypted. Having such a block in the encrypted domain, we can build a number of protocols for sorting, finding the minimum/maximum of a set of encrypted values and thresholding (Sections 3.4.2, 4.2.3 and 5.4.2). Even though the idea behind the comparison block is the same, the application and the setting require subtle changes in the resulting protocol. To explain these differences, consider the following tasks that involve comparison block:

- **Face detection.** The comparison block is used in a protocol which finds the minimum value of M encrypted distances by using binary-tree approach. The outcome of this is either *Yes* or *No*.

- **Clustering.** The same comparison block is used for finding the closest cluster to a user in an R-dimensional space. As in face recognition, K encrypted distances are compared in a binary-tree fashion. But in order to obtain the index of the cluster with the minimum distance, the protocol is modified such that each comparison outputs not only the minimum distance but also its index. The pointer to the closest cluster is kept secret both from the server and the user during the several iterations of the protocol and revealed only to the user in the end.

- **Recommender system.** The comparison block is used for obtaining an encrypted vector that consists of encrypted ones and zeros. Each element of this vector is an encryption of 1 if the corresponding similarity is above a public threshold and an encryption of 0 otherwise. The comparison block used in this application is redesigned to reflect two important changes: 1) one of the inputs of the comparison block is a publicly known value, i.e. a threshold and 2) the encrypted values to be compared with a threshold are packed in one encryption. These two factors resulted in a new comparison protocol which is less expensive in terms of computation and communication costs since data packing reduces the number of expensive randomization operations and the cost of communication. In addition, publicly known threshold allows us to realize part of the operations in clear, introducing a considerable gain in computation time.

As it is explained above, different applications and different settings require a different protocol even if the underlying operation is the same signal processing operation of comparison. Therefore, it is our conclusion that designing a single comparison protocol and plugging it to several applications is not always possible. Depending on which party has the data and the decryption key, what is public and what is private, the cryptographic protocol changes. We anticipate that other non-linear signal processing operations such as quantization may have a similar situation.

7.2.4 Data Expansion and Packing

As described previously, the bit length of data in multimedia applications even after scaling is much smaller than the cipher text space. Once the individual signal samples are encrypted, data expansion occurs which is usually in the order of hundreds. This data expansion introduces additional cost for data transmission and, if required, for storage.

To overcome this problem, we introduced packing values similar to [2, 11, 3]. If the application and the setting permit, packing values introduces a gain in the amount of data transmitted. As an example, in the recommender system scenario, instead of sending M encrypted values, the user sends only $R+1$ where $M \gg R$, due to packing $M - R$ values that are to be used for generating recommendations (Section 5.4.1).

Depending on the application, it may also introduce a reduction in the number of operations on the encrypted data. Considering the recommender system, the number of times secure multiplication protocol is run (Section 5.3.2) is reduced from R to 1 due to data packing. It is, however, notable that the cost of operations with the packed data may increase for a number of reasons like the need of using larger random values for blinding. Even in the case of an increase in the computational cost, data packing may still be useful for a computationally powerful player with limited bandwidth.

A drawback of data packing appears when unpacking is required during the processing. Unpacking is a costly, interactive protocol which consists of several decryptions [3]. For applications in which unpacking is necessary, the cost of this operation should be considered. Data should be packed in circumstances in which the gain by packing is more than the cost of possible unpacking operations.

7.2.5 Computational and Communication Costs

Data packing seems to be promising considering the communication cost as it introduces a considerable reduction on the amount of transferred data as discussed in previous sections. Even if the computational cost increases in some settings, data packing should be considered for applications with a limited bandwidth.

Considering computational costs, working in the encrypted domain is far more expensive than its counterpart in the plain domain. As an example, a clustering algorithm takes a few seconds whereas its privacy-preserving version can take an hour (Section 4.5). This is due to time consuming operations such as addition, multiplication and blinding in the encrypted domain. In addition to the time consuming operations, for non-linear operations we also need to invoke interactive cryptographic protocols. To reduce the computation cost, we investigated several options in this thesis:

- **Packing.** An effective way of reducing the number of operations in the encrypted domain is packing values when it is possible. By packing the number of operations in the encrypted domain reduces since instead of processing several encrypted values, we process less of them.

- **Precomputation.** A considerable amount of time is consumed for generating random values and randomizing encryptions. As it can be seen in Sections 3.7 and 4.5, generating random values and randomizing part of the encryptions prior to start of the program or in the idle time of the processor can reduce the time consumption significantly (by a factor of 3 in the case of secure clustering).

- **Optimizations.** Considering signal processing applications, redundant operations are hardly to be found. However, when we start working in the encrypted domain, there are several costly operations that can be circumvented. As an example, if values are added in the encrypted domain, instead of randomizing each value separately during the encryption, the sum can be randomized only once after the addition. An analysis of the implementation in depth can be rewarding.

7.3 Open Issues

In this thesis, we introduced principled solutions for preserving privacy in multimedia applications based on homomorphism and MPC techniques. As the idea of merging cryptography and signal processing is new, we investigated a number of challenges. Further progress can be made if the followings are considered.

Security level and homomorphic encryption schemes. We propose to preserve privacy by encrypting private data with a homomorphic cryptosystem. As we encrypt signal samples in multimedia applications, the number of encryptions is in vast amount. The encryption operations with currently available homomorphic encryption schemes are time consuming. Considering that the encryption also causes data expansion, using existing homomorphic cryptosystems is expensive. Cryptographers may find it interesting to analyze the sufficient level of security for protecting signal samples and develop a cryptosystem particularly for signal processing purposes.

Non-linear operations and MPC. The selected applications mostly consist of similar signal processing operations such as computing distances, correlation and thresholding. Other applications can be investigated to identify more operations. For instance, in order to implement compression techniques, quantization is crucial. Similar other operations should be considered in the context of secure signal processing.

Data packing. Packing has been used in this thesis and it proved its usefulness in efficiency regarding communication and computational costs. Yet, it has not been investigated in its full extent. For instance, scaling is possible if every value in the packed encryption is to be scaled with the same constant. The case of scaling with different factors is worth considering. In addition, existing unpacking operations can be studied further to reduce the complexity of the protocol.

Complexity. The run-time of the privacy preserving version of the applications presented in this thesis is promising yet, further research is necessary to deploy the cryptographic solutions for real use. While packing, precomputation and optimizations can be useful, a major breakthrough can be achieved if there was a practical fully homomorphic cryptosystem available. In such a case, expensive protocols that are repeated in vast amounts like secure multiplication protocol could be eliminated. Until such a cryptosystem is invented, reducing the computation cost of privacy-preserving multimedia applications depends mostly on better protocol design and optimization of the implementation.

Interaction. The cryptographic protocols for non-linear operations are interactive and require several rounds. In order to reduce the interaction, once again, a fully homomorphic cryptosystem is required.

Attacker model. Throughout this thesis, we proposed solutions based on semi-honest model. In this model, each player acts according to the protocol but keeps every exchanged message to deduce extra information. However, in real life this might not be the case. The proposed solutions should be considered for *active attacker* model in which the protocol should be robust against active attackers who may manipulate the protocol steps. In such cases, the required techniques for ensuring the correctness of the protocol are more resource demanding as additional cryptographic protocols like ZKPs for validating the actions are necessary. Such methods and their cost in malicious cases should be investigated.

Other cryptographic approaches. This thesis focuses on preserving privacy in multimedia application based on homomorphism and MPC techniques that are built over integer arithmetic. However, there is a large body of literature on the other MPC techniques:

- *Secret sharing.* Cryptographic protocols based on secret sharing [10] are studied in dept in literature. In a secret sharing scheme, a secret value v is shared among an arbitrary number of parties. Each party who is given a *share* of the secret, can not obtain the secret v from his share. Only if a sufficient number of parties combines their shares, the secret v can be reconstructed. Here, the sufficient number of parties depends on the scheme and the application.

As secret sharing schemes are often defined over integers, operations such as addition and multiplication by constant are significantly faster, even 'free' compared to equivalent operations in this thesis. However, other operations like multiplication of secrets require invoking protocols such as [4]. Moreover, as the data are shared between parties, the required storage space for an application with vast amount of users can be demanding. Secret sharing can be an option in secure signal processing however, further research is required to construct protocols for multimedia applications and to identify the limitations.

An alternative realization of secret sharing schemes is by using semantically secure homomorphic encryption schemes, assuming that we have a *threshold* variation such as the Paillier variant [5]. In this case, sharing is equivalent to distributing the encryptions of the secret and reconstruction to joint decryption. However, this alternative has the disadvantage of expensive joint decryption procedure in which sufficient number of users should provide input to the decryption.

- *Garbled circuits (GC).* As an MPC technique, garbled circuits focus on secure evaluation of Boolean functions. Any polynomial size Boolean circuit can be evaluated but rephrasing the application considered in this thesis as a Boolean circuit may not be feasible due to the size of the circuit. However, hybrid solutions that couple homomorphism and GC might be a good direction to investigate. Instead of rephrasing the whole application, a part of the protocol can be evaluated by GC while the other parts are realized using homomorphism and MPC techniques over integers. Recently, the hybrid idea has been used for preserving privacy for several scenarios [1, 6, 7, 8, 9], proving the effectiveness of this approach. Further research is necessary to use GC for secure signal processing.

7.4 Conclusion

Privacy is a severe concern among people who are using online applications. To prevent misuse of personal information and minimize the damage, several approaches are considered in the community. Firstly, people need to be educated about the possible threats. Educating people aims to increase the *awareness* of the people so that they can act responsibly when they are online. Secondly, *the law and the regulations* need to be updated to reflect the recent changes regarding the online applications and prevent malicious behaviors. Unfortunately, this procedure takes time and not every threat is foreseeable in the virtual world. Because of this, appropriate actions can only be taken when the damage is done. Thirdly, we need to have *technological solutions* to protect the privacy of the individuals in the application level.

In this thesis, as a technological solution, we proposed a new idea: secure signal processing. This approach combines cryptography and signal processing during the construction of the applications instead of applying cryptographic tools on top of the application. As the field is very new, it contains many challenges. Only a part of these problems have been addressed in this thesis. However, the proposed solutions for secure signal processing have shown that our approach is promising technological direction to protect the privacy of the users.

In order to continue further research in this direction, we have the following motivations. First, privacy concerns are increasing rapidly. This concern is growing as more 'technological applications' such as surveillance systems, taxing systems based on miles driven, health care systems and smart card applications are integrated to our lives. Second, regardless of the context of the application, the operations are from the signal processing field and thus, there are many fundamental operations in the design of such applications. Results obtained from the research on secure signal processing can be deployed in several other applications with certain modifications. For instance, the setting and the proposed solution for secure face detection is not very different than secure biometric identification or audio fingerprint detection.

References

[1] M. Barni, P. Failla, R. Lazzeretti, A. Paus, A.-R. Sadeghi, T. Schneider, and V. Kolesnikov. Efficient privacy-preserving classification of ECG signals. In *First IEEE Workshop on Information Forensics and Security*, pages 91–95, 2009.

[2] T. Bianchi, A. Piva, and M. Barni. Composite signal representation for fast and storage-efficient processing of encrypted signals. *IEEE Transactions on Signal Processing*, 2009.

[3] T. Bianchi, T. Veugen, A. Piva, and M. Barni. Processing in the encrypted domain using a composite signal representation: pros and cons. In *IEEE International Workshop on Information Forensics and Security*, pages 176–180, 2009.

[4] R. Cramer, I. B. Damgård, and U. Maurer. General secure multi-party computation from any linear secret-sharing scheme. *Lecture Notes in Computer Science*, pages 316–334, 2000.

[5] I. B. Damgård and M. J. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. *Lecture Notes in Computer Science*, pages 119–136, 2001.

[6] V. Kolesnikov, A.-R. Sadeghi, and T. Schneider. Improved garbled circuit building blocks and applications to auctions and computing minima. In *8th International Conference on Cryptology And Network Security (CANS '09)*, Lecture Notes in Computer Science. Springer, December 12-14, 2009. Full version available at `http://eprint.iacr.org/2009/411`.

[7] A. Paus, A.-R. Sadeghi, and T. Schneider. Practical secure evaluation of semi-private functions. In M. Abdalla, D. Pointcheval, P.-A. Fouque, and D. Vergnaud, editors, *ACNS*, volume 5536 of *LNCS*, pages 89–106, 2009.

[8] A.-R. Sadeghi and T. Schneider. Generalized universal circuits for secure evaluation of private functions with application to data classification. In *11th International Conference on Information Security and Cryptology (ICISC 2008)*, pages 336–353, Seoul, Korea, 3-5 December 2008. Springer-Verlag.

[9] A.-R. Sadeghi, T. Schneider, and I. Wehrenberg. Efficient privacy-preserving face recognition. In *12th International Conference on Information Security and Cryptology (ICISC'09)*, Lecture Notes in Computer Science. Springer, December 2-4, 2009. Full version available at `http://eprint.iacr.org/2009/507`.

[10] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[11] J. R. Troncoso-Pastoriza, S. Katzenbeisser, M. U. Celik, and A. N. Lemma. A secure multidimensional point inclusion protocol. In *ACM Workshop on Multimedia and Security*, pages 109–120, 2007.

List of Abbreviations

MPC	Multi-Party Computation
RSA	Rivest-Shamir-Adleman
TTP	Trusted Third Party
ZKP	Zero-Knowledge Proof
DCT	Discrete Cosine Transform
EM	Expectation Maximization
LUT	Look-Up Table
SVD	Singular Value Decomposition
WCA	Watermark Certification Authority
ZKWD	Zero-Knowledge Watermark Detection
DGK	Damgård-Geisler-Krøigaard
PCA	Principal Component Analysis
PC	Personal Computer
PDA	Personal Digital Assistant
PSP	Privacy Service Provider
BER	Bit-Error Rate
DC	Direct Current
DWR	Document-Watermark Ratio
KT	Kuribayashi-Tanaka
RDM	Rational Dither Modulation
PDF	Probability Density Function
QIM	Quantized Index Modulation
DC-QIM	Distortion Compensated QIM
SD-QIM	Subtractive Dither QIM
GC	Garbled Circuits

List of Symbols

$E_{key}(\cdot)$	Encryption function
$D_{key}(\cdot)$	Decryption function
id	Buyer identity
Id	The index of the cluster that a user is assigned to
S_{WCA}	Digital signature of WCA
$\text{sim}_{A,B}$	Similarity value for the users A and B
a, b	Values to be compared
$\mathbf{A}$	Approximation matrix of $\mathbf{P}$
$b^{(i)}$	Direction of comparison
$\mathbf{B}/b$	A binary string of hidden information/sample
c	Cipher text (Chapter 3), scaling factor (Chapter 6)
$c_{j,n}$	n^{th} element of cluster centroid j
c_i	Value computed for $\hat{d}$ and $\hat{r}$
C	Covariance matrix
C_i	i^{th} bit of addition of D and r
C_j	Cluster centroid
d	Dither (Chapter 6), randomized z (Chapter 3)
$\hat{d}, \hat{r}$	Private inputs to be compared
D	Dataset (Chapter 2), packed differences (Chapter 5)
D_i	Distance between the feature vector of the input image and i^{th} feature vector
$D_{i,j}$	Distance between i^{th} user and j^{th} cluster
$\mathbf{D}_k$	Personalized decryption LUT for client k
e_i	Masked c_i

$\mathbf{E}$	Long term master encryption LUT
g, h	Generators of the cryptosytems
$\mathbf{G}$	Gradient
$\mathbf{G}_k$	User contribution to the current gradient
h	Impulse response sample
$\mathbf{I}$	Identity matrix
k	Bit length of scaled ratings
K	Number of clusters (Section 2.3, Chapter 4), number of selected eigenvalues (Chapter 3)
$\mathbf{K}$	Binary signal as a key(Section 2.2.4)
ℓ	Bit length of distances
L	Size of LUTs (Chapter 2), number of the most similar users (Chapter 5)
m	Plain text
M	Number of samples (Chapter 2), images (Chapter 3), items (Chapter 5)
n	RSA modulo (Chapter 3)
N	RSA modulo (Chapter 2), size of the training images (Chapter 3)
p, q, u	Prime numbers
pk_A	Public key of A
$\mathbf{P}$	Preference matrix
P_i	User i represented as a point
$p_{i,n}$	n^{th} element of user point i
$Q_\Delta(.)$	Uniform quantizer with step size Δ
r	Random value
R	Dimension of space (Chapter 4), number of items to be used for similarity computations (Chapter 5)
$\mathbf{R}, \mathbf{S}$	Vector of random values
R_i, R_i'	Vectors of random values
s	Direction of the comparison
sk_A	Secret key of A
S_1	The number of α_i
S_2	The number of σ_i
s_j, n_j, α_j	Alice's shares
t_j, m_j, β_j	Bob's shares
t	Key size in bits
T	Threshold
T_1	The number of $x_{i,j}$ that can be packed in one encryption for computing α_i
T_2	The number of $x_{i,j}$ that can be packed in one encryption for computing σ_i
u_i	Eigenvectors
$x_{i,j}$	Elements of the preference vector V_i

$x'_{i,j}$	Elements of the preference vector V'_i
x'_i	Encrypted feature
$\mathbf{X}/x$	Signal/sample
$\mathbf{X}_j^{sum}$	Element j of the vector sum of X_i's
$\mathbf{X}''$	Encrypted and permuted watermarked content
$\mathbf{X}'$	Received signal (Section 2.4)
$\mathbf{X}'$, $\mathbf{Y}'$	Randomized vectors (Section 2.3)
$\mathbf{X}_w$	Watermarked content of vector $\mathbf{X}$
$\mathbf{Y}/y$	Signal/sample
Y_i	User i input matrix
Y'_i	Randomized Y_i
$Y_{j,n}^{sum}$	Element j, n of the matrix sum for Y_i's
$v(.)$	Function to normalize coefficients for RDM.
$\mathbf{V}$	Watermark
V_i	Preference vector for user i
V'_i	Scaled and rounded preference vector for user i
$\bar{w}_i$	Weights in $\bar{\Omega}$
w_j	XOR of $\hat{d}_i$ and $\hat{r}_i$
$\mathbf{W}_k$	Personalized watermark for client k
$\mathbf{W}/w$	Normally distributed random signal (watermark)
z	Difference of a and b
$\tilde{z}$	Comparison result
$\mathbb{Z}$	Set of integers
$\mathbb{Z}_N$	Residue class ring modulo N
$\mathbb{Z}_N^*$	Prime residue class group modulo N
α	DC-QIM factor
α_i	Packed preference vector for the recommendation generation
γ	Scaling factor
Γ	Input image
Γ_A	Binary vector of comparison result
Γ_{sum}	Sum of ratings for the items of interest
Γ_i	Binary vector output of comparisons for user i
Δ	Quantization step size
Θ	Training image
κ	Security parameter
λ	Comparison result of the private inputs
$\tilde{\lambda}$	Comparison result of the private inputs, direction randomized

λ'	Masked comparison result
Λ	Value composed of comparison results (Chapter 4), packed user vector for the recommendation generation (Chapter 5)
μ_j	Cluster centroid
ρ	Gain factor
$\rho_{\mathbf{X}'\mathbf{Y}}$	Correlation between $\mathbf{X}'$ and $\mathbf{Y}$
σ_w^2	Variance of the watermark
σ_n^2	Variance of noise
$\sigma'_j A$	Packed preference vector for the similarity computation of user A
$\sigma_{i,j}$	Sum of comparisons on the binary tree for $D_{i,j}$
$\sigma(\cdot)$	Secret permutation (Section 2.4)
Σ_A	Packed similarity values for user A
Φ	Training image, mean subtracted (Chapter 3), ratings vector of items of interest (Chapter 5)
Ψ	Average of the training images
Ω	Feature vector
$\bar{\Omega}$	Feature vector of the input image

Summary

Recent advances in technology provided a suitable environment for the people in which they can benefit from online services in their daily lives. Despite several advantages, online services also constitute serious privacy risks for their users as the main input to algorithms are privacy sensitive such as demographic information, shopping patterns, medical records, etc. While traditional security mechanisms can eliminate a number of attacks from outside, these mechanisms can not protect the privacy of the users as the service provider itself constitutes the biggest potential risk.

In this thesis, we focus on principled solutions to protect the privacy of users in multimedia applications. For this purpose we propose to keep the privacy-sensitive data safe by means of encryption during processing. This approach eliminates the risk of possible privacy abuse as the sensitive data is only available to the owner but no other party. However, once encrypted, the structure in data is destroyed as a consequence of the encryption procedure and thus we need appropriate tools to process encrypted data. Therefore, we focus on a number of cryptographic tools such as homomorphic encryption schemes and multiparty computation (MPC) techniques to realize privacy-preserving multimedia applications. The proposed principled solutions consider the signal processing aspect of the multimedia applications which is a new idea to the best of our knowledge.

In particular, we focus on a number of prototypical applications namely, face detection, user clustering in a social network, recommendation generation and anonymous fingerprinting. Based on these selected applications, we addressed the major challenges for secure signal processing: data representation, data expansion, realizing linear and non-linear operations and efficiency of the proposed protocols in terms of communication and computational costs. We propose to scale and round the signal values prior to encryption as these operations are highly inefficient to be realized in the encrypted domain. Moreover, we reserve sufficient space in terms of bit length for each signal sample to accommodate the possible expansion in bit size in the subsequent processing steps. However, reserving more bits for signals does not contradict with the data expansion problem. As the cipher text space is much larger than the size

of the original – and even scaled – signal samples, data expansion after encryption increases data transmission and storage costs significantly. In order to minimize the cost we propose to pack a number of signal samples in one encryption and process them when they are in the packed form. This approach requires cryptographic protocols particularly designed for the packed data but in the end saves considerable resources regarding bandwidth and storage capacity, even computational power.

Homomorphism plays a crucial role in our proposed solutions. With the help of homomorphic encryption, we are able to implement linear operations such as correlation and projection without interaction. However, linear operations are only a part of the signal processing. For the non-linear operations like distance computation, thresholding and comparison, we exploit MPC techniques. These techniques are often interactive and computationally expensive compared to the original systems in plain. However, by using data packing and designing the protocols with care, the communication and computational costs were reduced significantly.

In this thesis, we have shown that preserving privacy for multimedia signal processing is feasible. We determined the major challenges of secure signal processing and combined a set of cryptographic tools successfully with signal processing to realize the applications in the encrypted domain. The proposed solutions demonstrate that the privacy concerns in multimedia signal processing applications can be coped with by using cryptographic tools. Moreover, protocols that are designed to realize certain operations in the encrypted domain can be used in other applications and settings with a number of modifications.

Samenvatting

Recente vooruitgang in de technologie heeft een geschikte omgeving voor de mensen gecreëerd waarin zij kunnen profiteren van online diensten in hun dagelijks leven. Ondanks verscheidene voordelen, leveren online-diensten ook ernstige risico's op voor de privacy van hun gebruikers omdat de belangrijkste input voor algoritmen privacy-gevoelig is, zoals demografische gegevens, winkelgedrag, medische dossiers, enz. Terwijl er traditionele mechanismen bestaan tegen een aantal aanvallen van buitenaf, kunnen deze mechanismen de privacy van de gebruikers niet beschermen omdat de dienstverlener zelf het grootste potentiële risico vormt.

In dit proefschrift richten we ons op voorgestelde oplossingen om de privacy van gebruikers in multimedia-applicaties te beschermen. Voor dit doel stellen we voor om de privacy-gevoelige gegevens te beveiligen door middel van versleuteling tijdens de verwerking. Deze aanpak elimineert het risico van mogelijke schending van de privacy doordat de gevoelige gegevens alleen beschikbaar zijn voor de eigenaar, maar niet voor andere partijen. Echter, na versleuteling is de structuur in de gegevens vernietigd als gevolg van de versleuteling procedure en dus zijn geschikte instrumenten nodig om versleutelde gegevens te verwerken. Daarom richten we ons op een aantal cryptografische instrumenten zoals homomorphische encryptie schema's en multiparty computation (MPC) technieken om privacy te behouden in multimedia-applicaties. De voorgestelde oplossingen nemen het signaalverwerkingsaspect van de multimedia-toepassingen in acht, hetgeen een nieuw idee is, voor zover wij weten.

In het bijzonder richten we ons op een aantal prototypische toepassingen, te weten: gezichtsdetectie, gebruiker clustering in een sociaal netwerk, het genereren van aanbevelingen en anoniem fingerprinten. Op basis van deze geselecteerde toepassingen, snijden we de grootste uitdagingen voor veilige signaalverwerking aan: data representatie, gegevens expansie, het realiseren van lineaire en niet-lineaire operaties en de efficiëntie van de voorgestelde protocollen op het gebied van communicatie en computationele kosten. Wij stellen voor om de signaalwaarden voorafgaand aan de codering te schalen en af te ronden, daar deze operaties zeer inefficiënt zijn om te realiseren in het versleutelde domein. Bovendien reserveren wij voldoende ruimte in termen

van bit-lengte voor elk signaal sample om de mogelijke expansie in bit grootte in de verdere verwerking stappen te accommoderen. Echter, het reserveren van meer bits voor signalen is niet in tegenspraak met het gegevens-expansie probleem. Omdat de cijfertekst ruimte veel groter is dan het formaat van de originele –en zelfs geschaalde– signaal samples, verhoogt gegevens-expansie na versleuteling de datatransmissie- en opslagkosten aanzienlijk. Met het oog op het minimaliseren van de kosten stellen wij voor om een aantal samples van het signaal in te pakken in een encryptie en te verwerken in de verpakte vorm. Deze aanpak vereist cryptografische protocollen met name ontworpen voor verpakte gegevens, maar levert uiteindelijk aanzienlijke besparingen op ten aanzien van bandbreedte, opslagcapaciteit, en zelfs rekenkracht.

Homomorfisme speelt een cruciale rol in onze voorgestelde oplossingen. Met de hulp van homomorphische versleuteling zijn wij in staat om lineaire operaties uit te voeren, zoals correlatie en projectie zonder interactie. Echter, lineaire operaties zijn slechts een deel van de signaalverwerking. Voor de niet-lineaire operaties zoals afstandsberekening, drempelmethode en vergelijking, benutten we MPC technieken. Deze technieken zijn vaak interactief en computationeel duur in vergelijking met de oorspronkelijke systemen in klare tekst. Echter, met behulp van gegevens-verpakking en het met zorg ontwerpen van de protocollen, worden de communicatie- en computationele kosten aanzienlijk verminderd.

In dit proefschrift hebben we aangetoond dat het behoud van privacy voor multimedia signaalverwerking haalbaar is. Wij hebben de grote uitdagingen van veilige signaalverwerking aangewezen en met succes een reeks van cryptografische instrumenten met signaalverwerking gecombineerd om de toepassingen in het versleutelde domein te realiseren. De voorgestelde oplossingen tonen aan dat de privacy-aspecten in multimedia signaalverwerkings toepassingen kunnen worden veiliggesteld door gebruik van cryptografische middelen. Bovendien tonen we aan dat protocollen die zijn bedoeld om bepaalde operaties te realiseren in het versleutelde domein kunnen worden gebruikt in andere toepassingen en gebieden, met een aantal wijzigingen.

Acknowledgements

Like everyone else, I owe my progress not only to the people that I have worked with in the last four years but also to those that I have ever known in my life. Every person I know has some ingredients in what I am now. Saying that, I still feel responsible to mention at least some of the names here.

It was all my next door friend Öner's idea to apply for a Ph.D. position abroad. When there appeared an opportunity, it was Prof. Dr. Nadia Erdoğan who encouraged me to start this journey. Supported by my former master thesis supervisor, Prof. Dr. Bülent Örencik, I came to Delft to start my next challenge; leaving many, many good friends back in Istanbul: Orhan, Serdar, Hasan, Hüseyin, Peyman, Cüneyd, Tolga, Ender, Yusuf, Tacettin . . .

From the very first day, I was warmly welcomed to the ICT group. Umut, Hasan, Ronald, Mark, Peter Jan and Jacco were the firsts that I met. Then the rest of the big ICT family came. Discussions during the coffee break, LAN parties, group trips with my colleagues were always fun. I thank Anja, Ben, Saskia and Robert for their support during all those years. During their master projects, I had the chance to work with Jeroen and Ilyaz who could be very successful researchers but chose to work in the industry. I must say that I learned a lot on 'Dutch' humor thanks to Jeroen. And of course my office mates from whom I also learned a lot on life and research. Thank you very much Alper, Bartek, Katherina, Kosmas and Michael.

During my study in Delft, I was involved in a European Project, SPEED, with six partners from four countries. I had the chance to know Stefan Katzenbeisser, Jorge Guajardo, Ahmad-Reza Sadeghi, Bart Preneel, Gregory Neven, Mauro Barni, Alessandro Piva, Jamshid Shokrollahi, Frederik Armknecht and Martin, Christophe, Christian, Mina, Li, Antoon, Riccardo, Tiziano and Pierluigi. As a part of the collaboration I visited Philips High Tech Campus on a regular basis for a year and had the chance to know Berry Schoenmaker and Tomas Toft with whom I worked later in Aarhus, Denmark, during my research visit in fall 2009. During my visit in Aarhus, I met impressive researchers and good friends in the cryptography group of Ivan Damgård: Jesper, Martin, Marcel and Mikkel.

I would like to thank to Thijs Veugen and the members of the cryptography reading group for the nice discussions we had for the last couple of years. Especially discussions I had with Thijs always improved my work further.

Of course, everything for the last four years was possible thanks to Inald. I am honored of being his student. It was shocking to address a professor with his first name, though. During these four years I learned a lot from him. And it seems that learning keeps going when he is around.

I can not forget to mention my best friends here: Ozan, Zülküf and Beste. And constantly changing international community in Delft. I had four years full of everything with you. Thanks a lot!

What is the meaning of life if you don't share your happiness and sadness with others, especially with your family? Even though, they don't share my concerns about my research progress, meetings with my supervisor or paper deadlines, they were always there to support me. My dear sister, Lâle, uncle Mustafa and their adorable children Selin and Hakan; I am quite lucky that I have had you here for all these years. Last but not least my parents. I dedicate this thesis to my mother and to the memory of my father.

Curriculum Vitæ

Zekeriya Erkin was born in Nizip, Turkey on December 17, 1980. He graduated from Nizip Anatolian High School in 1998. He obtained B.Sc. and M.Sc. degrees from Computer Engineering Department in Istanbul University of Technology (ITU) in June 2002 and January 2005, respectively. In February 2006, he started as a Ph.D. candidate in the Information and Communication Theory Group, Delft University of Technology.

During his study in high school, he successfully represented his school in Ankara as one of the best schools of the country in 1998. Same year, he was ranked in the first thousand among 1.5 million students in the national university exam. After finishing his bachelor degree, he started as a research and teaching assistant in Computer Engineering Department of ITU where he represented R&T assistants in department and faculty boards from 2004 to 2006. During his Ph.D., he was involved in SPEED project, guiding master students and assisting laboratory sessions.

His interest areas involve cryptographic protocols, secure multiparty computation techniques, watermarking, fingerprinting and steganography. He is a member of ASCI and IEEE.